12-2018

Design and Implementation of Intelligent Guidance Algorithms for UAV Mission Protection

Karina Rivera

DESIGN AND IMPLEMENTATION OF INTELLIGENT GUIDANCE ALGORITHMS FOR UAV MISSION PROTECTION

A Thesis

Submitted to the Faculty

of

Embry-Riddle Aeronautical University

by

Karina Rivera

In Partial Fulfillment of the

Requirements for the Degree

of

Master of Science in Aerospace Engineering

December 2018

Embry-Riddle Aeronautical University

Daytona Beach, Florida

DESIGN AND IMPLEMENTATION OF INTELLIGENT GUIDANCE ALGORITHMS

FOR UAV MISSION PROTECTION

by

Karina Rivera

A Thesis prepared under the direction of the candidate's committee chairman, Dr. Hever Moncayo, Department of Aerospace Engineering, and has been approved by the members of the thesis committee. It was submitted to the School of Graduate Studies and Research and was accepted in partial fulfillment of the requirements for the degree of Master of Science in Aerospace Engineering.

THESIS COMMITTEE

Chairman, Dr. Hever Moncayo

Member, Dr. Richard Prazenica

Member, Dr. Snorri Gudmundsson

Graduate Program Coordinator, Dr. Magdy Attia — Date 12/5/18

Dean of College of Engineering, Dr. Maj Mirmirani — Date 12/5/18

Senior Vice President for Academic Affairs and Provost, Dr. Lon Moeller — Date 12/5/18

ACKNOWLEDGMENTS

It is with complete joy and gratitude for me to thank my mother Rosita, for always placing my brother and me first, for her unconditional love, for taking all the risks to fulfill our dreams, for her patience, support and motivation. Her strength to overcome any challenge, her charisma towards any unfortunate event, her integrity and perseverance in her work are what guide me to be the woman and professional I aspire to be.

To my brother Pato, for all the times he has shared his knowledge with me, from how to tie my shoes to how to integrate an equation, for all his love towards his younger sister, and is unconditional support. His curiosity and passion towards robotics and his career is what motivates me to be a better engineer.

I would like to thank my advisor Dr. Hever Moncayo, for the opportunity of being part of his research team, for all his patience, advice, and encouragement. He taught me that there is always a solution, no matter the challenge.

Thank you to my committee members Dr. Richard Prazenica, and Dr. Snorri Gudmundsson, for their invaluable guidance, knowledge and advice towards the realization of this thesis.

To the irreplaceable friendship I have made with Yomary, Juan, and Diana. Thank you for brightening my days, for all the cherished moments shared during our studies, for the assistance and motivation to finish my thesis.

Last but definitely not least, I would like to thank Nolan Coulter, for his blind believe in me and my capabilities, for all his support and motivation, for opening his heart and letting me be part of his life, and for sharing so many unforgettable moments.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS

AIS	Artificial Immune System
AP	Attitude Performance
ARP	Angular Rates Performance
DCM	Direction Cosine Matrix
DOF	Degrees of Freedom
FDI	Fault Detection and Identification
HM	Health Monitoring
IMU	Inertial Measurement Unit
ISL	In-situ Landing
NLDI	NonLinear Dynamic Inversion
NS	Negative Selection
PC	Power Consumption
PI	Performance Index
PWM	Pulse Width Modulation
RBP	Remaining Battery Percentage
SBC	Starting Battery Capacity
TRH	To-return Home
TT	Trajectory Tracking
UAV	Unmanned Aerial Vehicle

ABSTRACT

Rivera, Karina MSAE, Embry-Riddle Aeronautical University, December 2018. Design and Implementation of Intelligent Guidance Algorithms for UAV Mission Protection.

In recent years, the interest of investigating intelligent systems for Unmanned Aerial Vehicles (UAVs) have increased in popularity due to their large range of capabilities such as on-line obstacle avoidance, autonomy, search and rescue, fast prototyping and integration in the National Air Space (NAS). Many research efforts currently focus on system robustness against uncertainties but do not consider the probability of readjusting tasks based on the remaining resources to successfully complete the mission. In this thesis, an intelligent algorithm approach is proposed along with decision-making capabilities to enhance UAVs post-failure performance. This intelligent algorithm integrates a set of path planning algorithms, a health monitoring system and a power estimation approach. Post-fault conditions are considered as unknown uncertainties that unmanned vehicles could encounter during regular operation missions. In this thesis, three main threats are studied: the presence of unknown obstacles in the environment, sub-system failures, and low power resources. A solution for adapting to new circumstances is addressed by enabling autonomous decision-making and re-planning capabilities in real time.

1. Introduction

During the past few years, UAVs have gained attention due to their efficiency and reliability to perform a vast number of tasks that vary from civil to government applications such as package delivering, reconnaissance of unknown terrain, mapping capabilities for research or commercial purposes, traffic monitoring, flood mapping, law enforcement surveillance, in-situ atmospheric monitoring, and more. In addition, UAVs popularity is continuously growing due to advances in computer technology, software development, global navigation, sophisticated sensors, and more. With this motivation, the aerospace community has invested a significant amount of effort in solving common challenges associated with the effects that disturbances such as wind, failures in the system, battery drain, static and dynamic obstacles could have on the overall mission performance. Solutions offered by the industry to address these problems include fault tolerance control, adaptive control, neural networks, nonlinear dynamic inversion (NLDI) control, and others (Nguyen, 2009, Perez, 2015, Nguyen, 2006, Nguyen, 2008, Krishnakumar, 2012, & Ghaffar, 2016). All these control methods address the recovering potential of the system under disturbances and all the aspects on “how to control” a vehicle but do not address tactical decision-making aspects of “how to fly” the vehicle (Krishnakumar, 2002).

UAV post-fault capabilities are rarely investigated; therefore, few methods consider the post-fault status of the system and the different possibilities to complete the desired mission based on the remaining resources (Cetin, 2014, Cesare, 2015). Acknowledging the system’s post-fault conditions, an increasing number of studies are investigating online mission replanning and decision-making capabilities (Chamseddine, 2015, Yao, 2015, Hernandez, 2017, Ramirez, 2016, Moon, 2013, Kaneshige, 2005,

Boskovic, 2004), which are two of the most important features of intelligent algorithms (Pu, 2013, Jones, 2006, Lin, 2010, Jung, 2002, Atkins, 2012). For instance, Chamseddine et.al. acknowledged that not many fault-tolerant control strategies establish a relationship between the remaining resources and the post-fault conditions of the system. In their research, they proposed a reconfigurable fault-tolerant control with online decision-making and replanning capabilities for improving the quadrotor trajectory tracking performance under the presence of actuator faults.

The demand for the development of intelligent algorithms is driven by the complexity of missions and challenges UAVs confront in the development phase (DeGarmo, 2004). Some of the issues UAVs face are safety (collision avoidance, weather conditions, system reliability), air traffic (navigation, air traffic management for missions in urban environments), and socio-economic (public acceptance, government investment). Therefore, to produce robust and reliable unmanned vehicles to address these problems, the system must be capable of responding to changing goals while taking actions to preserve the integrity of itself and its environment.

Notwithstanding, before implementing any intelligent algorithm, it is important to define which attributes are required of a system for it to be considered intelligent (Sankararaman, 2017, Krishnakumar, 2004, Krishnakumar,2017, Kivelevitch, 2015). In today's society, intelligence is identified as expertise, talent, schooling, IQ and social interaction. From a computational perspective, intelligence can be defined as system flexibility, adaptability, memory, learning, temporal dynamics, reasoning and the ability to manage uncertainties (Krishnakumar, 2004). The same is suggested by Krishnakumar in most of his work on intelligent systems, where he lists the following as characteristics

desired for an intelligent system:

- Learning
- Adaptability
- Robustness
- Improving efficiency
- Information compression
- Extrapolated reasoning

With these capabilities, an autonomous intelligent system would be able to complete any mission without human interaction in an optimal pattern under post-fault conditions.

In this thesis, a proposed approach generates an intelligent system that is able to adapt autonomously in a fast and efficient manner to different uncertainties by on-line sensing, on-line replanning, and online decision-making actions. The designed intelligent system considers the current vehicle's conditions and mission goals, and after the vehicle recovers from a threat, the system finds the most optimal solution to complete the mission efficiently. The threats considered are unknown obstacles in the mission, internal failures, and low battery percentage.

To find the best solution after failure recovery, the proposed algorithm incorporates three main modules that are for unknown obstacle detection, the health status of the system (i.e., whether the vehicle presents an internal failure or not) and energy management (i.e., the remaining battery to complete the mission). Each module has been implemented with different features to compensate for the threats previously mentioned. The obstacle detection module consists of an online path planner for obstacle avoidance along with an

obstacle detection mechanism. The health status module integrates an Artificial Immune (AI) approach to track the health conditions of the system. The energy management introduces an online estimator for the system's battery percentage.

With all these capabilities, the UAV becomes a complex system by acquiring different autonomy layers. Recent efforts have been focused on providing a taxonomy for intelligent systems in order to identify the level of autonomy a vehicle possesses. Kivelevithch (Kivelevithch), 2015 developed a set of parameters to characterize intelligent systems as follows: methodology, learning, adaptation, adaptation trigger, adaptation scope, consistency, data sources, data fusion, and manifestation. Using these parameters, the intelligent algorithm proposed in this thesis has been also classified (with blue highlight) in each category as shown in Table 1. In addition, a general schematic for the different layers of autonomy the vehicle will have after implementing the intelligent algorithm is shown in Figure 1.

Table 1 Taxonomy Properties and Level for Intelligent Systems

Property	**Related to**	**Levels**	**Explanation**
Methodology (METH)	Learning	NONE	The system has no capability of learning
		SUP	The system is capable of supervised learning
		UNS	The system is capable of unsupervised learning
		BOTH	The system is capable of both supervised and unsupervised learning
Learning (LEARN)	Learning, Information Compression	NONE	The system has no capability of learning
		PRM	The system is capable of only learning new parameters within a given structure of rules
		STR	The system is capable of learning new rules and decision structures
		BOTH	The system is capable of learning new rules, structures, and parameters
Adaptation (ADAPT)	Adaptability, Extrapolation	NONE	The system is incapable of adapting to changing environments or internal states
		INT	The system is capable of adapting only to changes in internal states
		EXT	The system is capable of adapting only to changes in external states

		BOTH	The system is capable of adapting to both internal and external states
Adaptation trigger (TRIG)	Adaptability, Extrapolation	NONE	The system is insensitive to any triggers to adapt
		REACT	The system only adapts as a reaction to state changes after they occur
		PREEM	The system is capable of preemptive reaction to state changes before they occur by forecasting their changes
		BOTH	The system is capable of both reactionary and preemptive triggers to adapt to changes
Adaptation scope (SCOPE)	Adaptability, Extrapolation, Learning	NONE	The system is incapable of adaptation
		PRES	The system is capable of adaptation within a certain prescribed set of behaviors
		FLEX	The system is capable of free adaptation
Consistency (CONS)	Robustness, Extrapolation	NONE	The system's behavior under various conditions is inconsistent
		BEHAV	The system's behavior under various conditions is consistent, but there is no guarantee on performance
		PERF	The system's performance under various conditions is consistent, but it may use different behaviors to achieve it
Data sources (DATA)	Information compression	NONE	The system has no data sources
		UNI	The system has a single modality
		MULTI	The system has several modalities of information sources. They may be collocated or from several locations
Data fusion (FUSE)	Information compression	NONE	The system performs no level of data fusion
		SINGLE	The system performs a single level of data fusion model
		MULTI	The system performs several but not all levels of the data fusions model
		ALL	The system performs all levels of the data fusion model
Manifestation (MANI)	General	ENTITY	The manifestation of intelligent system properties is at the entity level
		COLLC	The manifestation of intelligent system properties is at the collective level
		HYBRID	Some of the intelligent system properties manifest at the entity level and some at the collective level

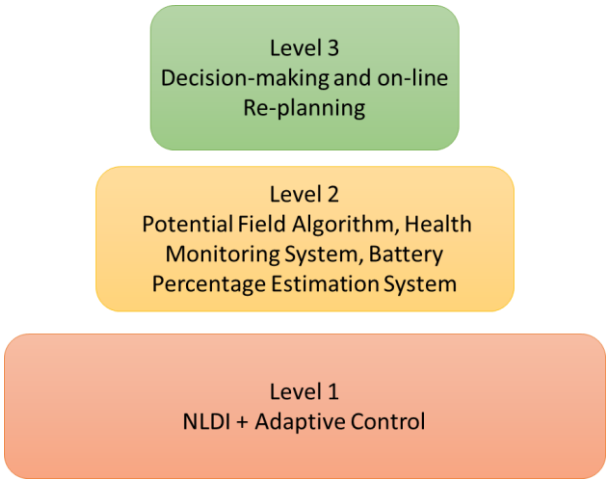

Figure 1 Autonomy levels for the proposed intelligent algorithm

The following chapters will provide a description of the design, development, and implementation of the proposed intelligent algorithm for a quadcopter test bed to achieve all autonomy levels shown in Figure 1. Chapter 2 addresses the overall design of the intelligent algorithm and the decision-making process. Chapter 3 presents two types of path planning algorithms used for trajectory generation. Chapter 4 describes the Artificial Immune System (AIS) approach implemented for monitoring in real-time the health of the vehicle. Chapter 5 describes the methodology for the vehicle power resources' estimation. Chapters 6 and 7 are dedicated to the simulation and implementation of the intelligent decision-making algorithms in the vehicle. Chapters 8 and 9 address some conclusions and future work.

This research effort has resulted in the following publications and submissions:

Accepted for Publication

Rivera, K., Moncayo, H., Verberne, J., Festa, D. (2019). "Design and Implementation of Intelligent Decision-Making Algorithms for Unmanned Aerial Vehicles Mission Protection". AIAA SciTech Forum. San Diego, California.

Verberne, J., Betancur, Y., **Rivera, K.**, Coulter, N., Moncayo, H. (2019). "Comparison of MRAC and L1 Adaptive Controllers for a Gimbaled Mini-Free Flyer". AIAA SciTech Forum. San Diego, California.

Journal Publication

Garcia, D., Perez, A., Moncayo, H., **Rivera, K**., Betancur, Y., DuPuis M, R., Mueller, P. (2017). "Spacecraft Health Monitoring Using a Biomimetic Fault Diagnosis Scheme". *AIAA Journal of Aerospace Information Systems (JAIS)*, Vol. 15, No. 7 (2018), pp. 396-413.

2. Intelligent Algorithms Decision-Making Methodology

The design and implementation of the proposed intelligent algorithm was studied under specific study cases for preliminary evaluation and demonstration of its capabilities.

2.1. General Architecture

To achieve intelligent navigation, the proposed system has been designed to possess the following capabilities:

1. The ability to generate optimal trajectories under normal and abnormal conditions. Path planning algorithms such as Clothoid and Potential Field have been implemented to calculate the most optimal trajectory for the UAV to complete the desired mission.
2. The capability of detecting unknown obstacles and identifying the magnitude of such obstacles. For this research, obstacles are considered as buildings or other vehicles and it is assumed that an onboard vision system would provide the obstacle's position and magnitude.
3. The system is able to detect and identify internal failures. Failure Detection and Identification (FDI) is achieved through the adoption of an Artificial Immune System (AIS) approach to monitor the health of the vehicle.
4. The system should be capable of estimating the power resources (in this case, the battery percentage) and making decisions based on that information. Through the implementation of a battery estimation model, the battery percentage is calculated. If the battery percentage goes below a desired threshold, the system will re-plan the mission with two possible outcomes: in situ landing (ISL), or to return home (TRH) maneuver.

These capabilities provide the level of autonomy to generate the optimal solution

after it has recovered from a threat. In order to evaluate and confirm the system is capable of such detection, identification, replanning, and decision-making potentials, the system is exposed to threats such as unknown obstacles in the path, failures in the system (i.e. such as the saturation of one motor), and low battery conditions.

Vision system, health monitoring, and power estimation are specific modules within the decision-making architecture that deliver the system's status information. This information is analyzed by a set of conditions that determine whether the mission needs to be replanned or not. If the mission needs to be replanned a trigger signal is sent to the path planner algorithm so an optimal solution can be generated. Figure 2 provides an overview of the proposed general architecture to follow a decision-making process. Figure 3 shows a more detailed representation of the modules and their interactions with their respective set of conditions required for path planner activation.

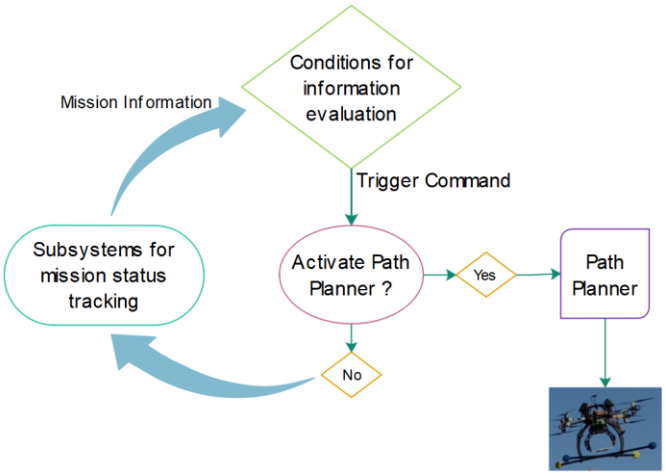

Figure 2 General architecture of the decision-making process to achieve an intelligent system

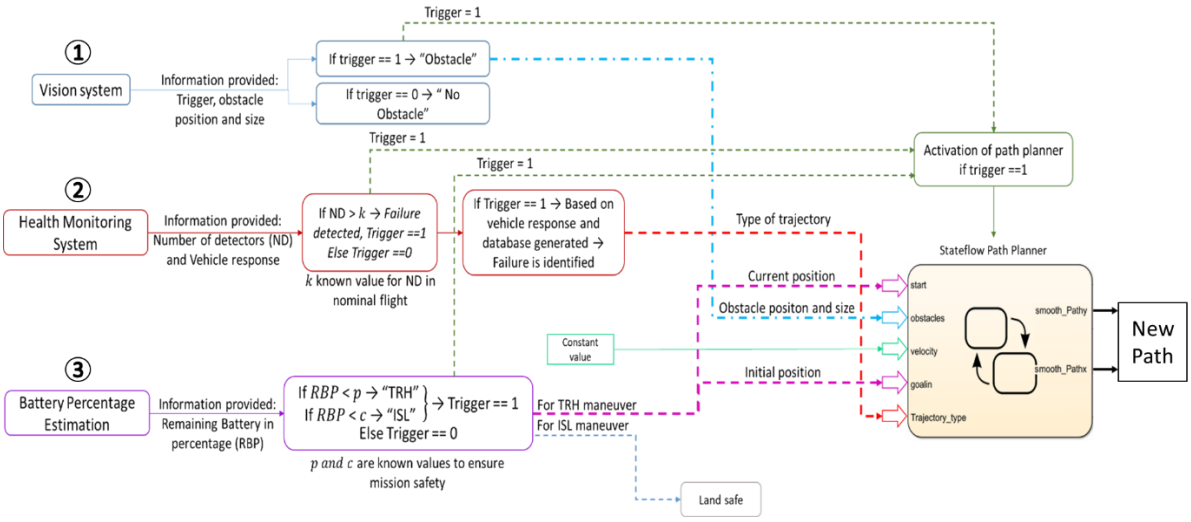

Figure 3 Subsystems and conditions required for path planner activation

As shown in Figure 3, the path planner algorithm generates a new trajectory based on a set of inputs such as the start position of the mission, obstacles positions (if known), the velocity of the vehicle, the desired goal of the mission and the trajectory type. Each of these inputs change accordingly depending on the type of failure and the module that is active. For instance, if an unknown obstacle is detected, the vision system will give new information for the obstacles position. Therefore, only the path planner input "obstacle position and size" will change. The health monitoring module works in a similar way as the vision system. The only input that will change due to a failure will be the "type of trajectory". On the other hand, if the power estimation module depicts the system does not have enough battery to continue the mission and the best solution is to return home, then the path planner inputs such as "start position", "desire mission goal" will change to new parameters. The following subsections describe more in detail the overall process for decision-making based on the information retrieved from these modules.

2.1.1. Vision System Module

This first module is in charge of detecting any unknown obstacle in the mission. If an unexpected obstacle is identified the subsystem sends a trigger indicating the presence of such entity. This trigger signal is equal to one when a new obstacle is encountered; otherwise, the trigger signal is equal to zero indicating there are no unknown obstacles in the path. In addition to the trigger command, if the value of the trigger is equal to one, the module provides the x-y position of the new obstacle in earth reference frame along with the size of the object. The position of the obstacle is sent as an input for the path planner and the trigger signal activates the path planner generating a new optimal trajectory. In this case, the start position of the mission will change to the current vehicle position, and the obstacles position will add the new obstacle information.

2.1.2. Health Monitoring Module

The second module is the Health Monitoring (HM) module. HM is inspired by an artificial immune system approach that resembles the immune system of living organisms. In other words, this module functions in the same way as the defense mechanisms of living entities. For instance, when a person is ailing, the human immune system generates antibodies to fight the illness. Similarly, when there is a failure in the system (such as the loss of one motor) the health monitoring module detects the failure by activating the antibodies (or also known as detectors) generated. This activation informs the system of the existence of an off-nominal behavior by sending a trigger signal equal to one. In addition to detecting a failure, another feature of the HM module is the identification of the type of failure affecting the system. The type of failure has a direct relation to the type of trajectory the vehicle must follow to complete the mission in the most efficient way. This

information, the trigger command equal to one and the type of trajectory the vehicle must follow, is sent for the activation of the path planner and for trajectory replanning. Chapter 4 explains with further detail the HM process and how the detectors are obtained.

2.1.3. Battery Percentage Module

The third module in the decision-making process is the battery percentage estimation. This module estimates the remaining battery percentage using the mathematical expressions detailed in Chapter 5. Based on this information, if the remaining battery percentage (RBP) is less than a certain threshold, the system makes the decision of ISL or TRH and activates the path planner by sending a trigger signal equal to one. In this case, the start position of the mission will change to the current position of the vehicle and the goal position of the mission will change to the values used for the start position of the overall mission. For instance, if the start position is defined as the origin after the vehicle makes the decision of returning home due to low battery percentage, the vehicle goal position will be $x=0$ and $y=0$.

2.2. Performance Metrics

A set of performance metrics have been developed to evaluate the vehicle's performance under nominal and abnormal conditions between two types of trajectories (Garcia, 2017, Perez, 2016). These metrics quantify the degradation of the mission and provide a characterization for the type of trajectories that benefits the mission the most. The set of metrics to evaluate the vehicle response under different flight conditions include Trajectory Tracking (TT), Attitude Performance (AP), Angular Rates Performance (ARP) and Power Consumption (PC).

The TT performance metric represents how close the error between the commanded and actual positions is to zero. If the error is zero the performance metric is 1. This metric is defined as:

$$\tilde{e}_{TT} = 1 - \frac{1}{CTT}\left(\sqrt{\frac{1}{T}\int_0^T e_x^2\,dt} + \sqrt{\frac{1}{T}\int_0^T e_y^2\,dt}\right) \tag{1}$$

The AP metric is how close the error between the commanded and actual attitude angles is to zero, and it is defined as:

$$\tilde{e}_{\Theta} = 1 - \frac{1}{C\Theta}\left(\sqrt{\int_0^T e_{\phi}^2\,dt} + \sqrt{\int_0^T e_{\theta}^2\,dt} + \sqrt{\int_0^T e_{\varphi}^2\,dt}\right) \tag{2}$$

Similar to the previous metrics, ARP metric represents how close to zero the error is between commanded and actual rates, and it is defined as:

$$\tilde{e}_{\Delta\Omega} = 1 - \frac{1}{C\Delta\Omega}\left(\sqrt{\int_0^T e_{\dot{\phi}}^2\,dt} + \sqrt{\int_0^T e_{\dot{\theta}}^2\,dt} + \sqrt{\int_0^T e_{\dot{\varphi}}^2\,dt}\right) \tag{3}$$

Finally, the PC performance metric represents the amount of effort the vehicle actuators have to perform to maintain stability. Effort equal to zero is considered the best response resulting in a performance metric of $\tilde{m} = 1$. The Power Consumption Performance Metric is defined as:

$$\tilde{m} = 1 - \frac{1}{C\Delta M}\left(\sum_{i=1}^{8}\sqrt{\int_0^T M_i(t)\,dt}\right) \tag{4}$$

In Eq. 1 through Eq. 4 , $\frac{1}{CTT}$, $\frac{1}{C\Theta}$, $\frac{1}{C\Delta\Omega}$, and $\frac{1}{C\Delta M}$ are normalization factors based on

the worst performance case. The parameters e_i represent the error between commanded and actual response of the ith state, and M_i is the PWM (Pulse Width Modulation) of each motor. The combination of these metrics result in a Global Performance Index (PI) as shown in Eq. (5)

$$PI\_UAV = [w_1\tilde{e}_{TT} + w_2\tilde{e}_{\Theta} + w_3\tilde{e}_{\Delta\Omega} + w_4\tilde{m}] \quad (5)$$

The parameters w_1, w_2, w_3, and w_4 are arbitrary weights to determine the contribution of each metric. Therefore, the sum of these weights equals one. The range of PI_UAV in Eq. (5) varies from 0 to 1, and for analysis purposes, 1 is considered the best case where the error between the commanded and actual states is zero. Notice that a number below and close to 1 would not necessarily represent a degradation of the flight performance due to an abnormal condition but could also represent the inability of the control system to maintain zero attitude, zero trajectory tracking errors, and high control effort.

3. Path Planning Algorithms

Assuming a threat has been identified, a decision-making process takes place where the vehicle is commanded to follow a trajectory generated based on the current status of the mission. For this reason, path planning algorithms are used to generate the desired trajectory and re-plan a new trajectory if uncertainties disturb the system or current mission.

Path planning has acquired great popularity among autonomous systems due to its use to perform complex tasks and the higher autonomy level that it provides to UAVs. There are several methodologies for trajectory generation including discrete and pose-based methods (Min, 2015, Chang, 2016, Bhaduri, 2009, Allen, 2016, Tsourdos, 2011). For this research effort, only the Clothoid and Potential Field algorithms have been studied and evaluated for implementation within the decision-making architecture.

3.1. Clothoid Algorithm

The Clothoid algorithm has gained its popularity due to its capability of generating piecewise trajectories with continuous curvature. Apart from other algorithms, such as Dubins path planner (Wilburn, 2013, Jeyarman, 2005, Kikutis, 2017), Clothoid algorithm avoids sharp edges and commands trajectories that are continuous in acceleration and velocity (Scheuer, 1997, Shanmugavel, 2010). In addition, Clothoid algorithm generates the trajectory to pass through the desired waypoints that the user defines.

To calculate a continuous Clothoid path, it is necessary to calculate the functions that represent the curvatures of the trajectory. These Clothoid curvatures will ensure the trajectory is continuous (Montes, 2007). To calculate the curvatures, four coordinates systems are determined as shown in Figure 4

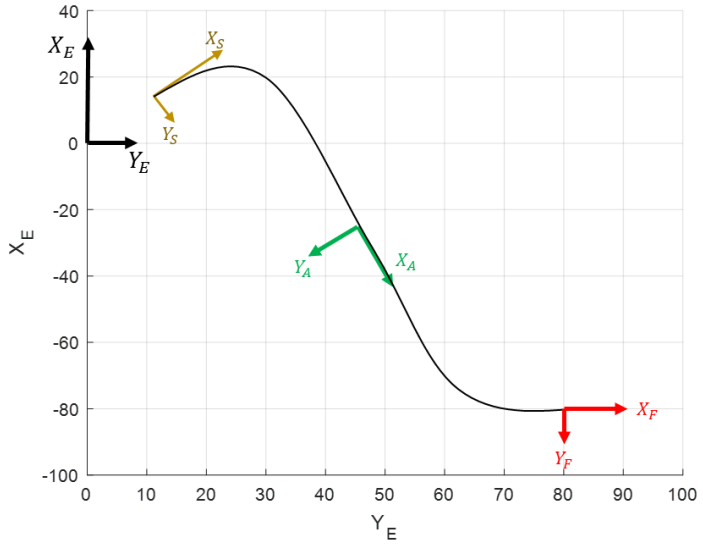

Figure 4 Coordinate System for Clothoid arcs calculations (Wilburn,2013)

$[X_E,Y_E]$ represent the Earth coordinate frame, $[X_S,Y_S]$ represent the start coordinate frame, $[X_A,Y_A]$ represent the connection coordinate frame and $[X_F,Y_F]$ is the final coordinate frame of the Clothoid path. In addition, the start and final positions are defined below, where X and Y are the positions in the respective coordinate frame, ψ is the heading orientation angle with respect to X_E, and $\varkappa$ is the maximum curvature which is defined by the UAV flight envelope capabilities.

$$P_S = [X_S\ Y_S\ \psi_S\ \varkappa_S] \tag{6}$$

$$P_F = [X_F\ Y_F\ \psi_F\ \varkappa_F] \tag{7}$$

As the final trajectory needs to be in the Earth reference frame, each coordinate frame is transformed using the following transformation matrices.

From start to Earth coordinate frame

$$R_{ES} = \begin{bmatrix} cos(\psi_S) & sin(\psi_S) \\ -sin(\psi_S) & cos(\psi_S) \end{bmatrix} \tag{8}$$

From final to Earth coordinate frame

$$R_{EF} = \begin{bmatrix} cos(\psi_F) & sin(\psi_F) \\ -sin(\psi_F) & cos(\psi_F) \end{bmatrix} \quad (9)$$

From connection reference frame to Earth

$$R_{EA} = \begin{bmatrix} cos(\psi_S + \phi_S) & sin(\psi_S + \phi_S) \\ -sin(\psi_S + \phi_S) & cos(\psi_S + \phi_S) \end{bmatrix} \quad (10)$$

The final set of the coordinate axes will be the basis for the generation of the Clothoid arcs, which are calculated from the solution of the Fresnel integrals presented in Eq. 11 and Eq.12 (Wilburn, 2013). The x and y are coordinates in the Clothoid axes, h is the length of the Clothoid arc and ϕ is the total sweep angle defined in Eq. 13 where q is a Clothoid curvature parameter.

$$x(h) = \int_0^h cos(\phi)\, dq \quad (11)$$

$$y(h) = \int_0^h sin(\phi)\, dq \quad (12)$$

$$\phi(q) = \frac{\kappa}{2h} q^2 \quad (13)$$

Through some equation manipulation explained more in detail in (Wilburn, 2013), the scaled Fresnel integrals are obtained, which are solved in order to generate the Clothoid curves and produce a piecewise continuous trajectory. Eq. 14 and Eq. 15 show the scaled Fresnel integrals.

$$C(h) = \sqrt{\frac{2h}{\kappa}} \int_0^{\bar{h}} cos(\bar{q}^2)\, d\bar{q} \quad (14)$$

$$S(h) = \sqrt{\frac{2h}{\kappa}} \int_0^{\bar{h}} sin(\bar{q}^2)\, d\bar{q} \tag{15}$$

After the Clothoid curves are calculated the algorithm generates continuous trajectories passing through desired poses defined by the user. Two examples of Clothoid trajectories are shown in Figure 5 and Figure 6. In both examples the start and end point is the origin, the light blue curves are the calculated Clothoid arcs and the circles represent the desired waypoints defined by the user.

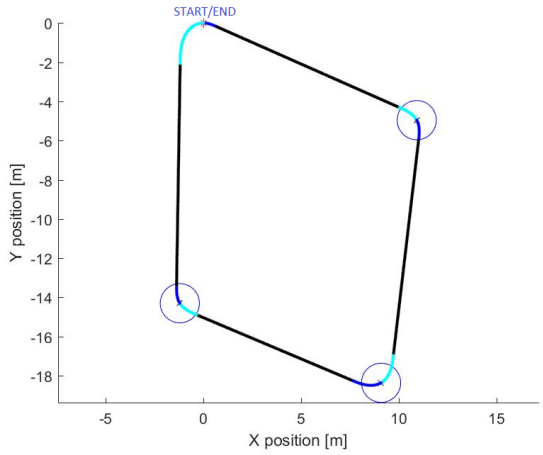

Figure 5 Diamond shape Clothoid trajectory

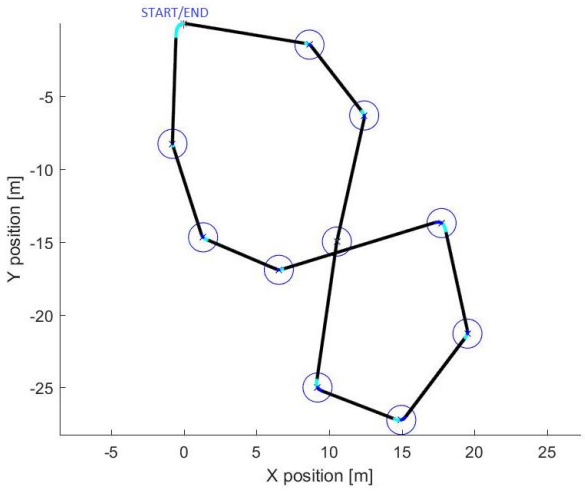

Figure 6 Eight shape Clothoid trajectory

3.2. Potential Field Algorithm

The well-known discrete method, potential field, is used to calculate trajectories and avoid obstacles (Hwang, 1992, Cruz, 2012, Cetin, 2016, Yuanchen, 2017, Paul, 2008). One of the advantages of using potential filed algorithm is the simplicity of the path planning solutions, which leads to low computational process requirements. On the other hand, the flaw of using this algorithm is the risk of falling into a local minimum, which can cause the algorithm to fail.

The principle behind the potential field relies on attractive and repulsive forces that generate the desired trajectory. The goal is treated as an attractive force whereas the obstacles act as repulsive forces guiding the vehicle through the environment. The vehicle is treated as a point mass, and the attractive and repulsive forces are defined as potentials as represented in Eq.16, where q is the current configuration of the vehicle.

$$\vec{F}(q) = -\nabla U(q) \qquad q \in \mathbb{R}^2 = \begin{bmatrix} x \\ y \end{bmatrix} \tag{16}$$

In Eq.16, $\vec{F}(q)$ is the total force acting on the UAV and $U(q)$ is the potential function at the configuration q. Thus, Eq. 16 can be decomposed in terms of attractive and repulsive forces and potentials as represented by Eq.17. As a rule, the attractive potential increases in magnitude as the vehicle steers away from the goal, and the repulsive potential increases in magnitude if the distance between vehicle and obstacle decreases.

$$\vec{F}(q) = \overrightarrow{F_{att}}(q) + \overrightarrow{F_{rep}}(q) = -\nabla(U_{att}(q) - U_{rep}(q)) \tag{17}$$

Once a definition for the total force acting on the UAV is obtained, the challenge is to select a potential function that satisfies the system conditions. Two functions are selected for attractive potential and one function for repulsive potential.

There are two common choices for attractive potential, one of these functions is expressed in Eq.18 where ρ_{goal} is $\rho_{goal} = \|q - q_{goal}\| = \sqrt{(x - x_{goal})^2 + (y - y_{goal})^2}$ and represents the distance between the current and goal positions. With this potential and Eq.19, which is a simplified definition of Eq.17, the attractive force is calculated as shown in Eq.20.

$$U_{att}(q) = \frac{1}{2}\xi\rho_{goal}^2(q) \tag{18}$$

$$\overrightarrow{F_{att}}(q) = -\nabla U_{att}(q) \tag{19}$$

$$\left[\overrightarrow{F_{att}}(q)\right]_E = \begin{bmatrix} -\xi(x - x_{goal}) \\ -\xi(y - y_{goal}) \end{bmatrix} \tag{20}$$

It is important to mention that the resulting force in Eq.20 is unbounded when the distance between the current and goal position increases. The second choice for the attractive potential is defined by Eq.21 and by some equations manipulation, the solution for the force is calculated as expressed in Eq.22.

$$U_{att}(q) = \frac{1}{2}\xi\rho_{goal}(q) \tag{21}$$

$$\left[\overrightarrow{F_{att}}(q)\right]_E = \begin{bmatrix} \dfrac{-\xi(x - x_{goal})}{\|q - q_{goal}\|} \\ \dfrac{-\xi(y - y_{goal})}{\|q - q_{goal}\|} \end{bmatrix} \tag{22}$$

Similar to the solution in Eq.20, the solution for the second potential function has a singularity when the vehicle is approaching the goal. Thus, the combination of Eq.20 and Eq.22 results in a general bounded solution for the attractive force as shown below, where

d is a condition parameter for ρ_{goal}.

$$\overrightarrow{F_{att}}(q) = \begin{cases} -\xi(q - q_{goal}), & \rho_{goal}(q) \leq d \\ \dfrac{-\xi(q - q_{goal})}{\|q - q_{goal}\|}, & \rho_{goal}(q) > d \end{cases} \tag{23}$$

Contrary to attractive potential, only one function is chosen for the repulsive potential. In a similar way as in Eq.19, a simplified version of Eq. 17 is defined for the repulsive potential. Therefore, the repulsive force is calculated as shown in Eq. 25. Eq.24 is a modified equation for the repulsive potential that solves the problem of a local minimum as mentioned before.

$$U_{rep}(q) = \frac{1}{2}\eta RI_i \left(\frac{1}{\rho_{obs}(q)} - \frac{1}{\rho_0}\right)^2 \rho_{goal}^n(q) \tag{24}$$

In Eq. 24 ρ_{obs} is the distance between the current position and the obstacle position taking into account R_{obs}, the obstacle radius. Thus, $\rho_{obs}(q) = \sqrt{(x - x_{obs})^2 + (y - y_{obs})^2} - R_{obs}$, and ρ_0 is the distance of influence of the obstacle.

$$\begin{aligned} \overrightarrow{F_{rep}}(q) = \eta RI_i \Bigg[& \left(\frac{1}{\rho_{obs}(q)} - \frac{1}{\rho_0}\right)^2 (q - q_{goal}) \\ & - \left(\frac{1}{\rho_{obs}(q)} - \frac{1}{\rho_0}\right)\left(\frac{\rho_{goal}(q)}{\rho_{obs}(q)}\right)^2 \left(\frac{q - q_{obs}}{\rho_{obs}(q)}\right) \Bigg] \end{aligned} \tag{25}$$

Through the implementation of these forces, a potential field is calculated which generates the optimal path while avoiding obstacles during the mission. An example of this algorithm is provided in Figure 7, where the red circles represent the radius of pre-defined obstacles, and the blue circles define a “warning zone” area for the vehicle. The area encountered between the red and blue circle indicates the presence of an obstacle.

Nevertheless, it is still safe to fly through this area without compromising the mission.

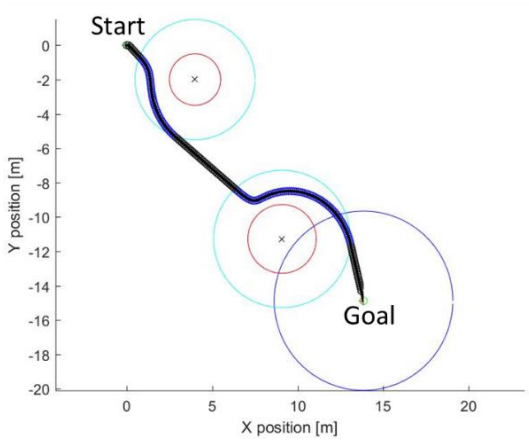

Figure 7 Example Potential Field Algorithm

With the potential field method, if a threat (failure, obstacle, etc.) is detected in the environment or within the system, the vehicle can make a decision based on current conditions and re-plan a new trajectory. In this way, the vehicle adapts to unexpected events.

Additionally, Potential Field algorithm can be enhanced to generate 3D trajectories. To equations to calculate the Potential Field in 3D follow the same process as the equations for 2D. Therefore in order to calculate the attractive force the following equation is used, where $q \in \mathbb{R}^3 = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$, and ρ_{goal}, the distance from the vehicle to the desired goal, is

$$\rho_{goal} = \|q - q_{goal}\| = \sqrt{(x - x_{goal})^2 + (y - y_{goal})^2 + +(z - z_{goal})^2}$$

$$\overrightarrow{F_{att}}(q) = \begin{cases} -\xi(q - q_{goal}), & \rho_{goal}(q) \leq d \\ \dfrac{-\xi(q - q_{goal})}{\|q - q_{goal}\|}, & \rho_{goal}(q) > d \end{cases} \quad (266)$$

Similarly, the same equation used for 2D is expanded to calculate the repulsive

force in 3D. In Eq.27 $q \in \mathbb{R}^3 = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$, ρ_{goal} follows the same representation as in Eq.26, and ρ_{obs}, the distance between the vehicle and each point of the 3D obstacle (sphere, cylinder), is represented as $\rho_{obs}(q) = \sqrt{(x - x_{obs})^2 + (y - y_{obs})^2 + (z - z_{obs})^2} - R_{obs}$.

$$\overrightarrow{F_{rep}}(q) = \eta R I_i \left[\left(\frac{1}{\rho_{obs}(q)} - \frac{1}{\rho_0} \right)^2 (q - q_{goal}) - \left(\frac{1}{\rho_{obs}(q)} - \frac{1}{\rho_0} \right) \left(\frac{\rho_{goal}(q)}{\rho_{obs}(q)} \right)^2 \left(\frac{q - q_{obs}}{\rho_{obs}(q)} \right) \right] \quad (277)$$

For simulation purposes two types of obstacles are designed: spherical and cylindrical. For all the 3D potential field trajectories generated, the green point represents the start and red point represents the end of the trajectory.

Spherical Obstacles

As mentioned, the Attraction Force calculation is based on the distance between the vehicle current position and the goal position. The Repulsive Force calculation is based on the distance between the vehicle position and each of the obstacles' position. To simulate the obstacle, the "*sphere*" MATLAB function is used which generates all the necessary points to form a sphere. Thus, the distance mentioned before will be calculated using the current vehicle position and each of the sphere points.

The results are based on the following coordinates:

$$Start\ position = [100\ 100\ 100]m$$

$$Obstacle 1\ position = [400\ 400\ 400]m$$

$$Obstacle\ 2\ position = [950\ 900\ 400]m$$

$$Obstacle\ 3\ position = [900\ 514\ 400]m$$

$$Goal\ position = [1200\ 1200\ 500]m$$

For simulation purposes, the radius of obstacles was determined to be $R = 100\ m$. Figure 8 through Figure 11 show different views of one example for 3D obstacle avoidance using the potential field approach. From these figures is seen that the vehicle is able to avoid all the obstacles.

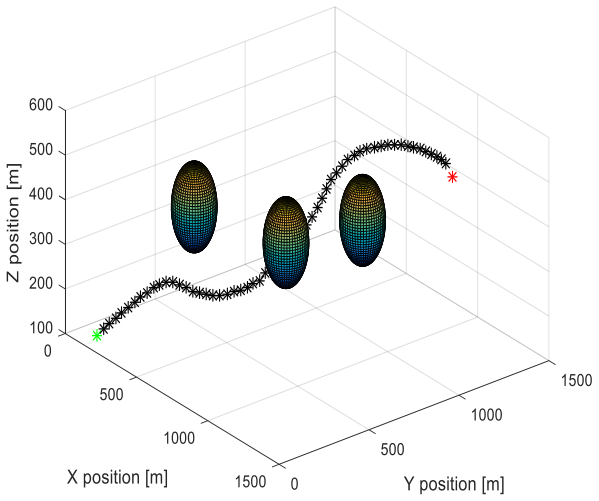

Figure 8 3D view of Potential Field performance

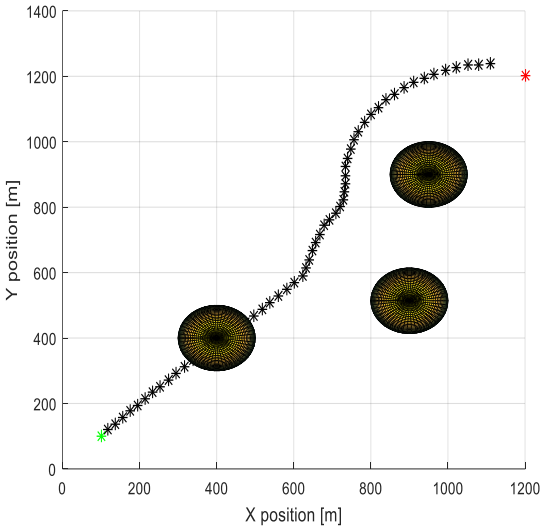

Figure 9 X-Y view of Potential Field Performance

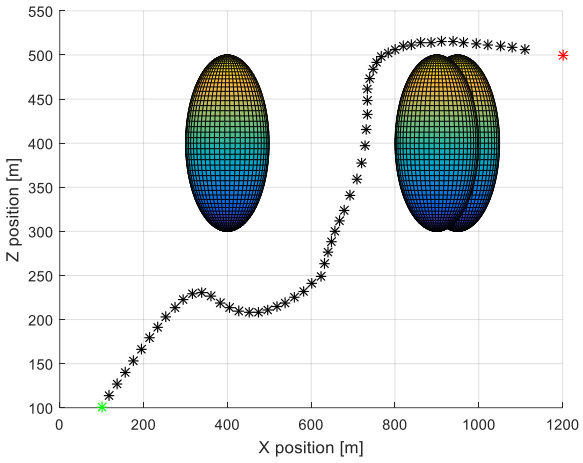

Figure 10 X-Z view of Potential Field performance

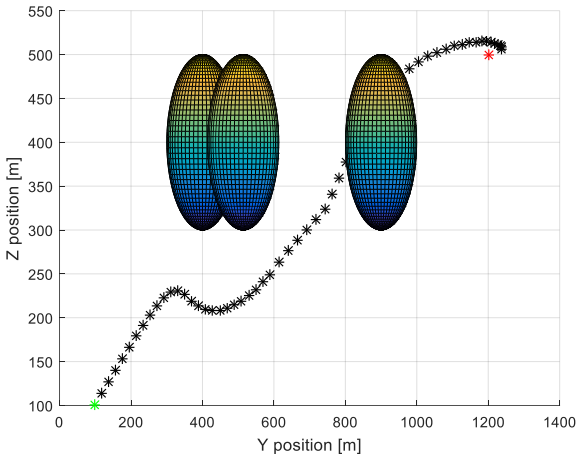

Figure 11 Y-Z view of Potential Field performance

Cylindrical Obstacles

Using a similar approach, the Attraction Force is calculated based on the distance between the current vehicle position and the goal position.

To calculate the Repulsive Force, the obstacles were simulated using the MATLAB function, "*cylinder*". Depending on the cylinder height the function provides the points only for the upper and bottom circles that form the cylinder. Then, using the bottom circle data and a condition of how many circles will be placed between the upper and bottom circles, additional circles were calculated to fill the cylinder. Using these cylinder points and the vehicle's current position, the Repulsive Force was calculated.

The results are based on the following coordinates:

$$Start\ position = [100\ 100\ 100]m$$

$Obstacle1\ position = [400\ 400\ 400]m$, Height=600 m

$Obstacle\ 2\ position = [950\ 900\ 400]m$, Height=900 m

$Obstacle\ 3\ position = [900\ 514\ 400]m$, Height=800 m

$$Goal\ position = [1200\ 1200\ 500]m$$

The radius for the cylinder was determined to be $R = 100\ m$

Figure 12 through Figure 15 show different views of a trajectory generated using the potential field 3D algorithm in a different environment. In all the figures, the start is represented as a green point and the end is pictured as a red point. This example could be more applicable to real life missions such as the exploration of urban environments. As in the previous example for cylindrical obstacles, the trajectory avoids all obstacles completing the mission safely.

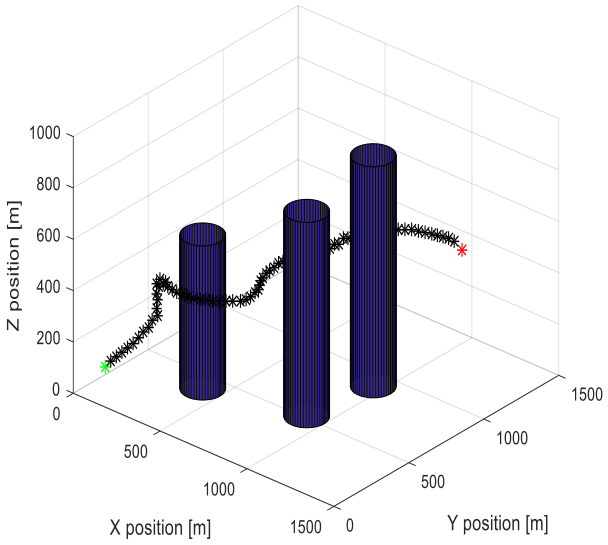

Figure 12 3D view of Potential Field performance

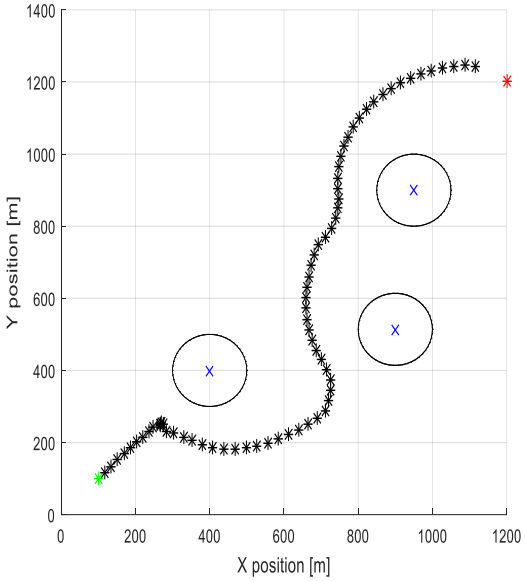

Figure 13 X-Y view of Potential Field performance

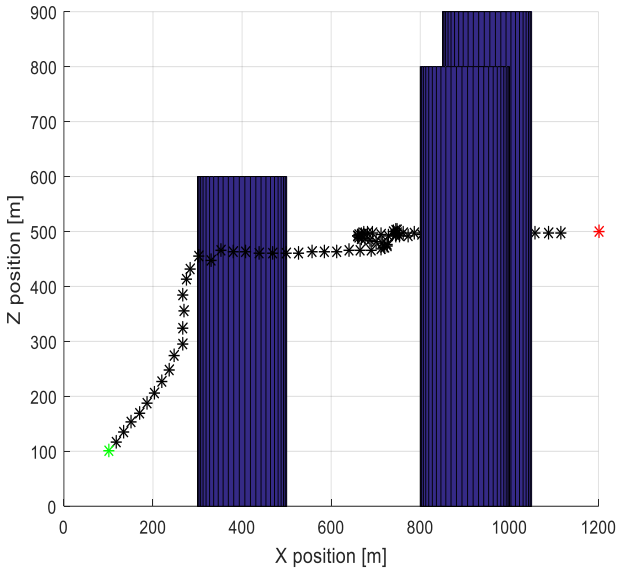

Figure 14 X-Z view of Potential Field performance

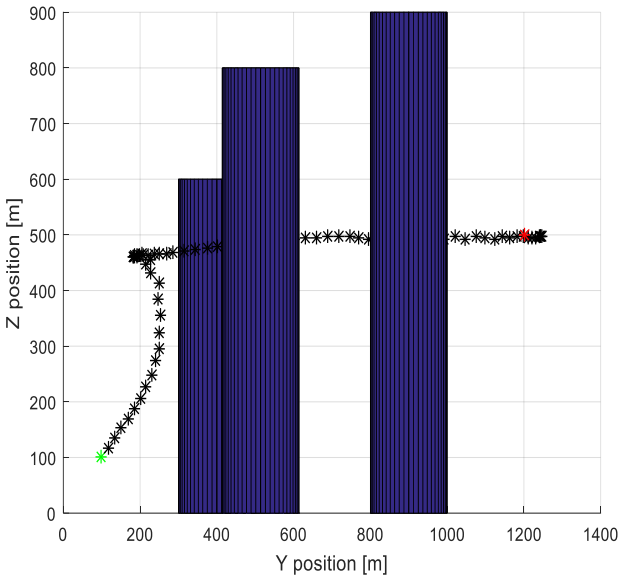

Figure 15 Y-Z view of Potential Field performance

The 2D Potential Field path planner is used in this thesis for on-line and off-line trajectory generation. Therefore, trajectories generated with potential field are commanded to the vehicle to follow, and if mission replanning is needed the same algorithm is used for replanning. Therefore, the potential field 2D is incorporated as the path planner within the decision-making process shown in Figure 3.

In this research effort, the Clothoid Algorithm is only used for off-line trajectory generation. One of the reasons is because potential field is easier to implement, satisfy the required capabilities for the design and development of an intelligent algorithm and is has less computational processing than Clothoid. However, it is possible to implement Clothoid algorithm for real-time path planning.

4. Health Monitoring

The Health Monitoring (HM) system is one of the main elements in the decision-making process. HM provides the current status of the vehicle's components including sensors, actuators, propulsion system, and more. Based on this information, the vehicle can be commanded to follow a new path to safely complete the mission if a failure in the system is detected.

In this thesis, an AIS approach has been adopted to perform real-time monitoring of the vehicle's health. AIS is a relatively new approach to fault detection and identification that is inspired by the biological immune system of living organisms (Kaneshige). The main idea behind HM is the generation of detectors and identifiers. The process to generate such elements is achieved by the implementation of a Negative Selection (NS) algorithm. NS is used for the generation of antibodies or detectors in a similar way as living organisms use it to eliminate intruders. NS relies on the self and non-self discrimination mechanism of the biological immune system. By performing a matching operation, if a specific feature does not match to the existing selves, which contain information of the nominal genetics of the system, then such feature is determined to be an intruder (Hongwei, 2009, Perez, 2015). This simple natural mechanism is used to generate antibodies and implemented through a design process as shown in Figure 16.

The process starts with post-processing data from flight tests, which captures the vehicle's nominal features from performing autonomous trajectory tracking missions. Due to the overlapping of a high number of features' samples, a clustering and data elimination is used for data reduction purposes. A set of self- projections are then built using these clusters which contain the finger prints of the dynamic response of the system under

nominal conditions. Following the negative selection approach, antibodies are then generated for each self-projection using a genetic based optimization algorithm to cover the non-self hyper-space. It is expected that during current operation of the UAV, if flight data samples fall within the non-self space, then an antibody or antibodies are likely to be activated and an abnormal condition can be declared. Then, based on the antibodies that are activated for that specific flight condition, such an abnormal condition could also be identified in terms of a failed sub-system or failure magnitude.

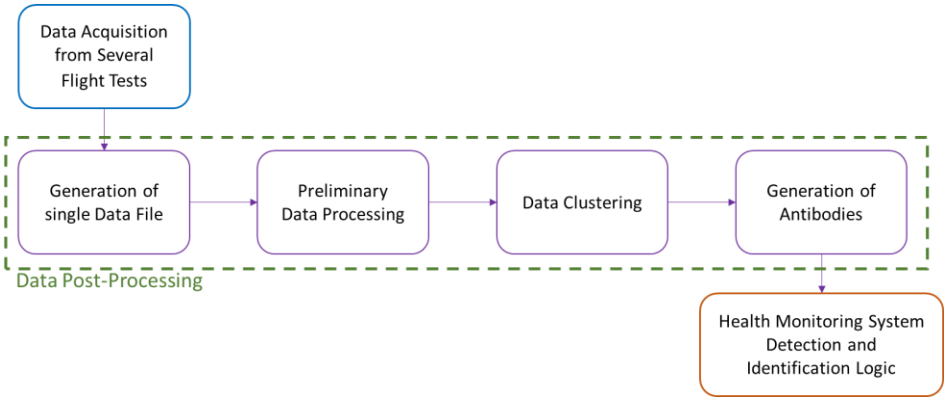

Figure 16 AIS for Health Monitoring of a UAV

After the detectors are activated, the identifiers are in charge of communicating to the system the type of failure the vehicle experiences. The success of the identification will rely on the quality of the selected identifiers to capture the dynamic signature of failures. Some of the parameters considered to be identifiers include the vehicle state variables, pilot input variables, variables generated within the control laws, etc (Perhinschi, 2010, Moncayo, 2010). These identifiers are labeled through an offline process using the detectors generated. For the case where a specific set of detectors, that include the parameters chosen to be identifiers are being activated, then the HM system informs the detection and identification of a certain failure. In this way, detectors can be labeled offline and their activation will indicate the type of failure.

Following the methodology proposed to create a health monitoring mechanism, several autonomous flight test under nominal and abnormal conditions have been performed. The features that capture the vehicle's response are obtaining by an off-line analysis of the flight tests data acquired. The selected features used for the generation of antibodies are shown in Table 2 and are combined in 2D projections (called selves), as shown in Table 3. There can be several combinations using the features in Table 2, but only the combinations with high data quality were chosen. In other words, the selves selected were the most effective for detection performance. The generated projections are a key element of failure detection and identification.

Table 2 Features used for the detection scheme

r_cmd	Yaw rate reference command
p_cmd	Roll rate reference command
$\emptyset\_cmd$	Roll reference command
p	Roll rate
q	Pitch rate
r	Yaw rate
$\emptyset$	Roll attitude
θ	Pitch attitude

Table 3 2D Selves projections for failures detection

Self #	**Features**	
1	p	r
2	q	r
3	r	$\emptyset$
4	r	θ
5	r	$\emptyset\_cmd$
6	$\emptyset\_cmd$	r_cmd
7	p_cmd	$\emptyset\_cmd$

If a failure occurs during the mission, the selves will not find a match with the selves generated off-line causing the activation of the antibodies (or detectors) that indicates a failure. The failure data will then fall outside the nominal data, causing the antibodies to identify a failure. Figure 17 to Figure 23 show each projection shown in Table 3. For all projections the blue points represent the nominal data acquired form flight tests. The red circles represent the generated antibodies that will identify a failure if the data goes outside the nominal blue data.

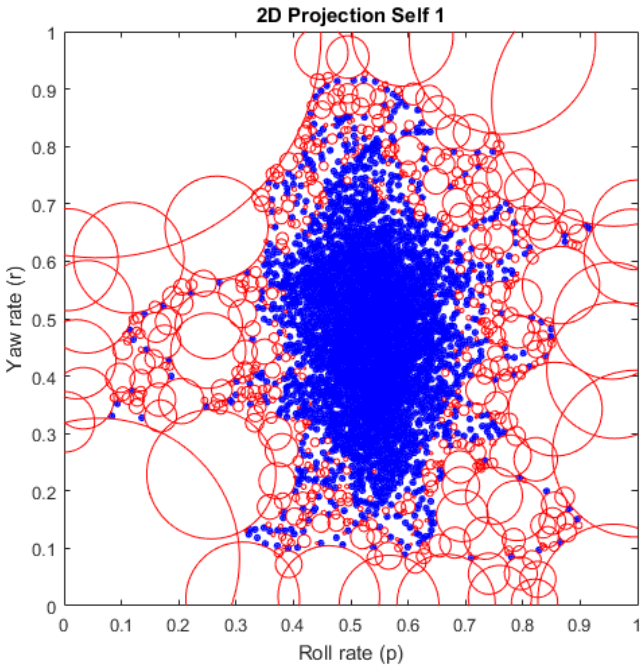

Figure 17 2D Projection Self #1 – roll rate vs yaw rate

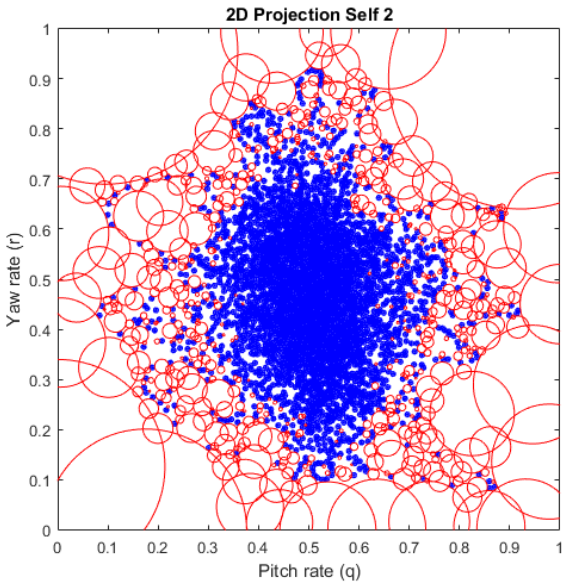

Figure 18 2D Projection Self #2 – pitch rate vs yaw rate

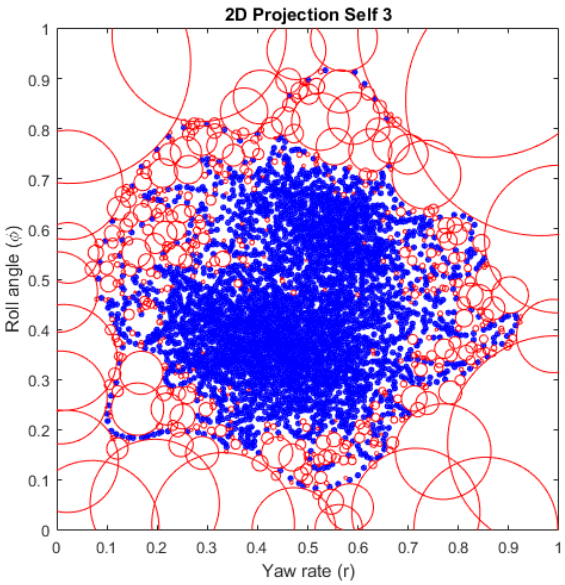

Figure 19 2D Projection Self #3 – yaw rate vs roll angle

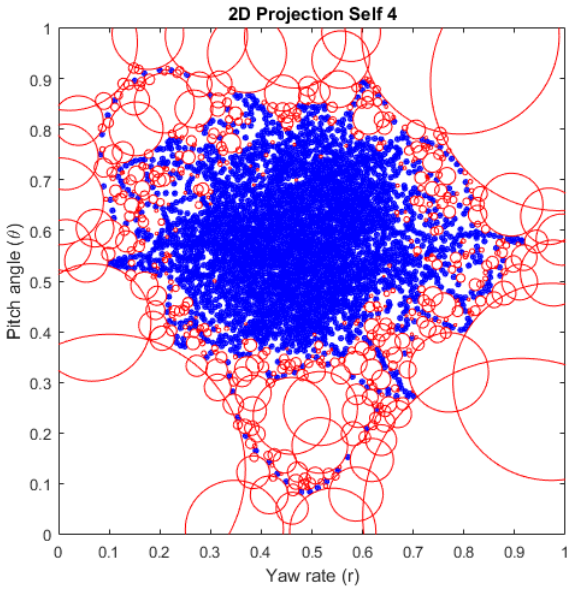

Figure 20 2D Projection Self #4 – yaw rate vs pitch angle

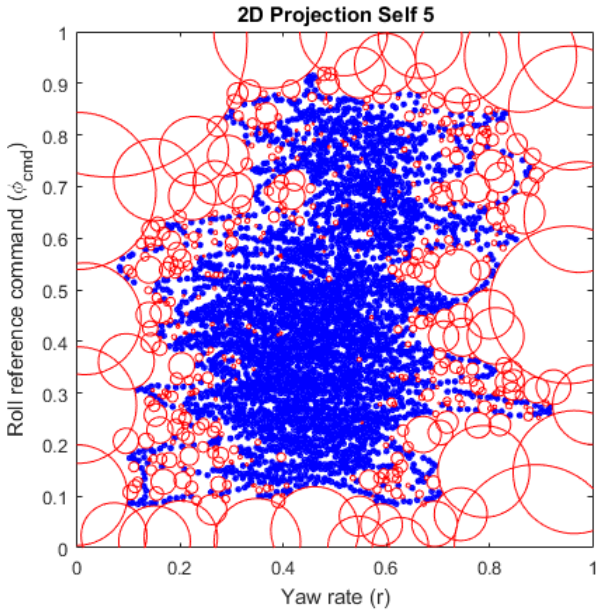

Figure 21 2D Projection Self #5 – yaw rate vs roll reference command

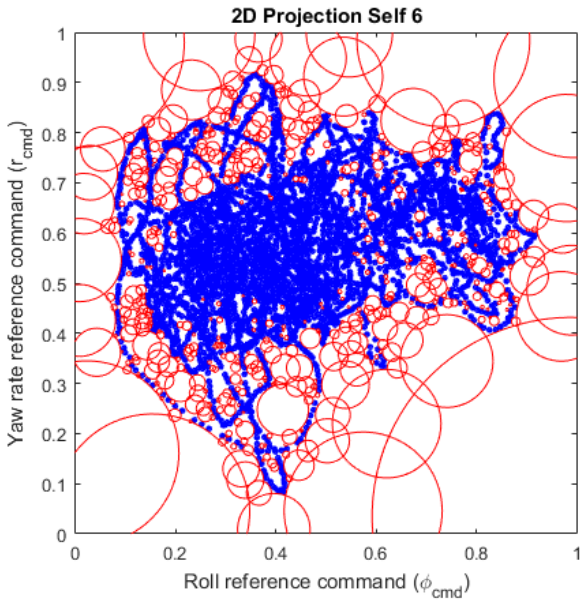

Figure 22 2D Projection Self #6 – roll reference command vs yaw rate reference command

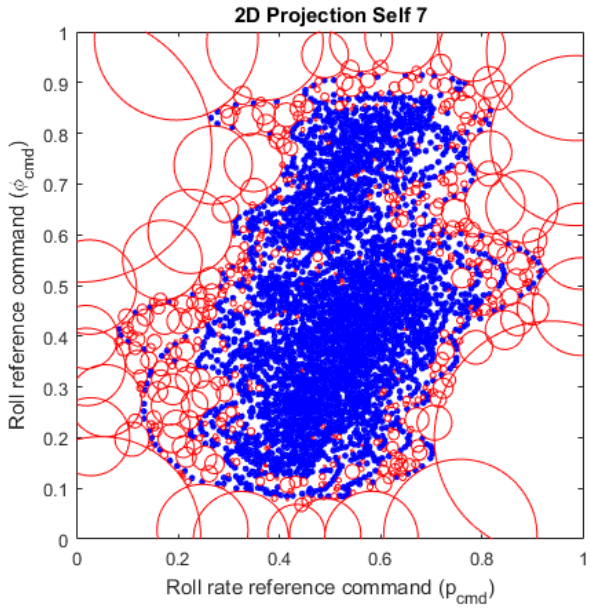

Figure 23 2D Projection Self #7 – roll rate reference command vs roll reference command

In Figure 24 results are presented to show the performance of the HM and antibodies activation under nominal condition. In this case, it is expected that a small amount of false alarms (antibodies activation) is produced. As an example, Figure 24a presents the PWM time history of one of the motors (x-axis represents time, y-axis represent the commanded PWM), with the other motors producing similar actuation to maintain commanded vehicle attitude and mission trajectory. Figure 24b shows a low amount of the total antibodies activation, which is translated to a low set of false alarms (x-axis represents time, y-axis represents the total activation of the antibodies). Figure 24c shows the individual activation history for each self (for each plot x-axis represents time, and y-axis represents the activation of the self). The activation of each self is passed through a buffer with a preselected size that monitors the activated antibodies over a specific window. Since there are seven selves, the maximum activation of the total antibodies is 70. For nominal cases, there should be zero activation but one of the reasons

for the presence of false alarms is due to uncertainties the HM cannot be trained offline, such as wind. Because wind is highly unpredictable, the vehicle will have a different response depending on the wind fluctuation.

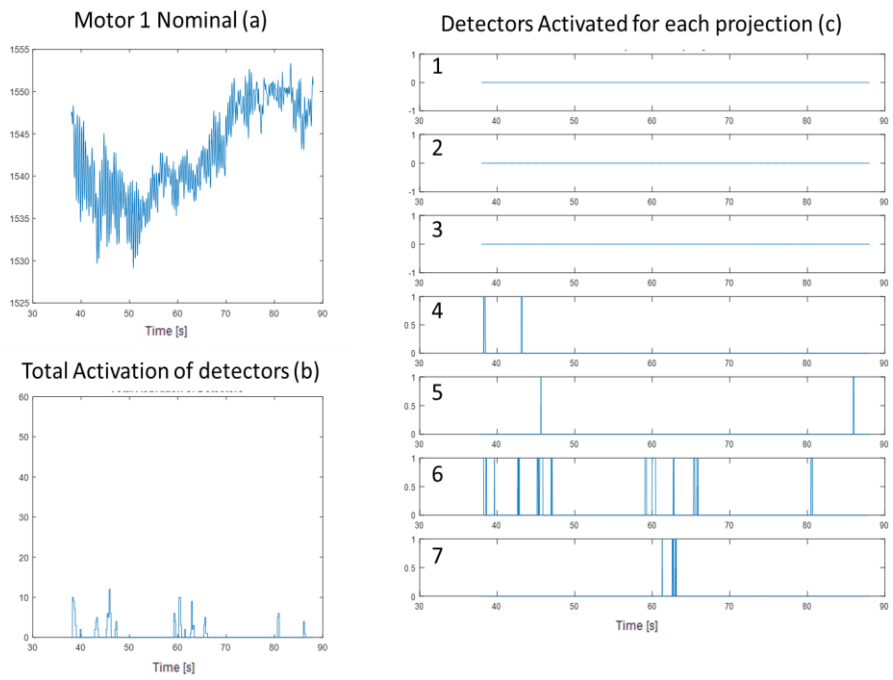

Figure 24 Vehicle health monitoring under nominal conditions

Similarly, HM logic was applied to a set of data for a flight test performed under failure conditions (saturation of one motor). Figure 25a shows PWM time history of the failed motor with a simulated saturation at around 51s. Figure 25b shows a high activation of antibodies after the failure has been injected. Figure 25c shows the activation of each self after the failure was injected, and shows the activation of the antibodies when the selves do not match with selves trained off-line.

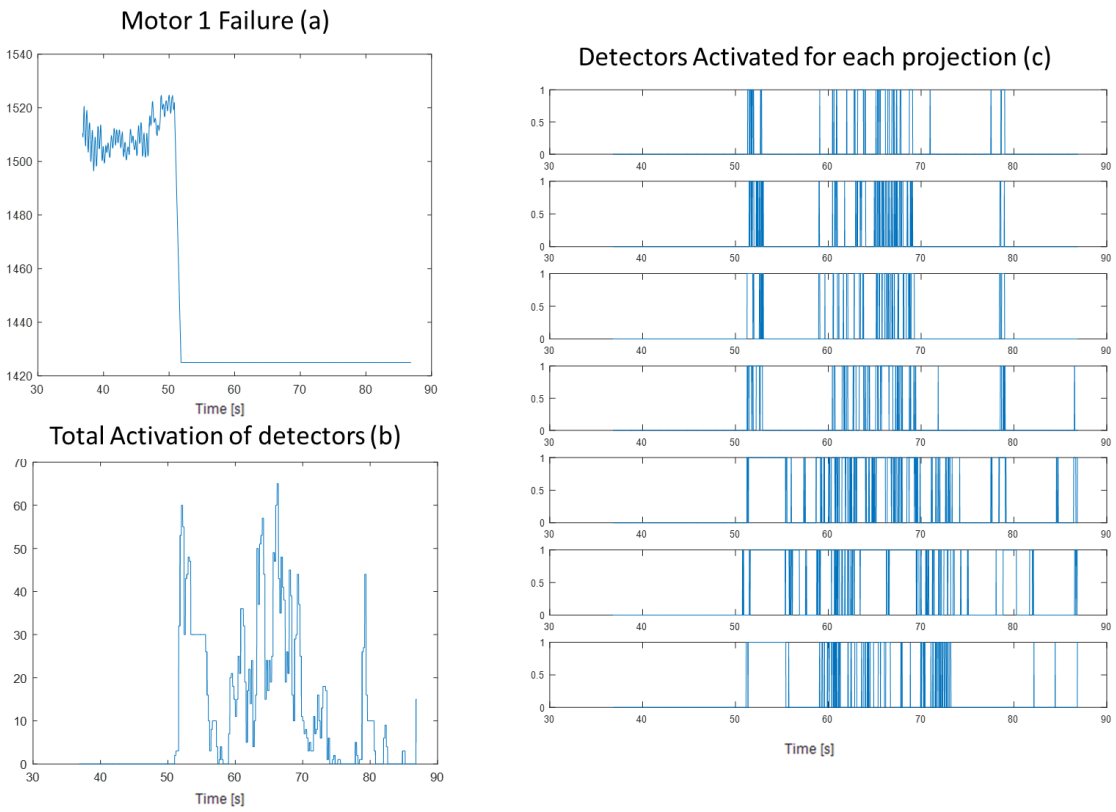

Figure 25 Vehicle health monitoring under abnormal conditions

In a similar way to identify the type of failure, the antibodies were trained offline for a specific projection, which is self 7. From flight tests data analysis it was identified that the self's 7 responses under different failures showed the highest contrast between each other. Therefore, this self is used to identify the type of failure the system is subject to. If this information is properly translated into a warning message which contains information about the detected and identified failure, then the intelligent algorithm can generate a safer trajectory to protect the overall mission.

5. Vehicle Power Resources Estimation

In order to inform the system the power status of the vehicle, an estimation method for the battery percentage has been developed, providing two solutions such as a “return to home” (RTH) maneuver and an “in-situ landing” (ISL) maneuver.

Figure 26 describes the process for the battery percentage estimation.

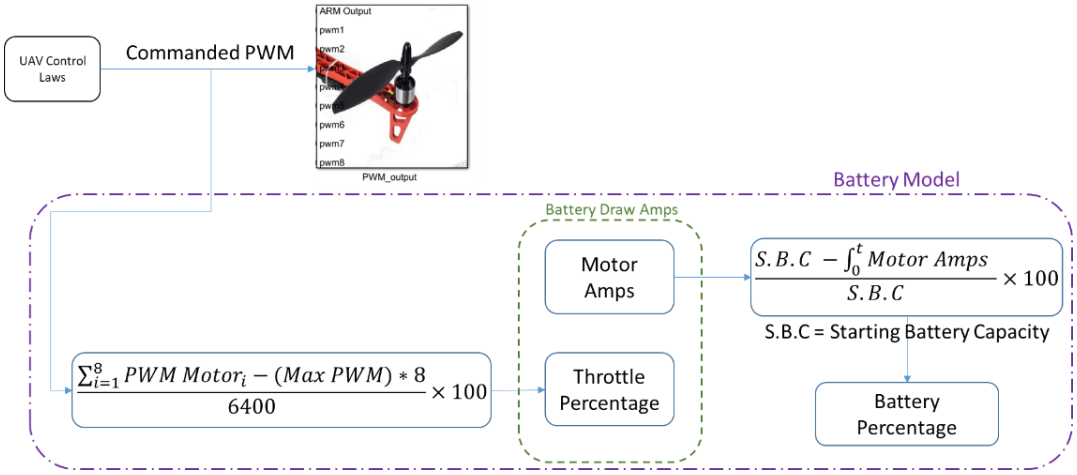

Figure 26 Remaining battery percentage estimation

The features used for the calculations are the commanded PWM sent from the onboard flight computer to each of the vehicle motors. From the PWM information, a throttle percentage is calculated which represents the amount of throttle the vehicle needs to perform desired maneuvers as shown in Eq.26.

$$\frac{\sum_{i=1}^{8} PWM\ Motor_i - (Max\ PWM) * 8}{6400} \times 100 \tag{28}$$

Based on this parameter and using the amperage draw from the motor’s data sheet at the current throttle percentage, an estimation of the battery-discharging rate is calculated and expressed in terms of percentage using Eq.27. This parameter represents the remaining battery percentage.

The starting battery capacity (SBC) is 97%.

$$\frac{S.B.C\ - \int_0^t Motor\ Amps}{S.B.C} \times 100 \tag{29}$$

Figure 27 shows the general architecture of the battery percentage estimation implemented in MATLAB/Simulink using the same scheme from Figure 26, Eq.26, and Eq.27.

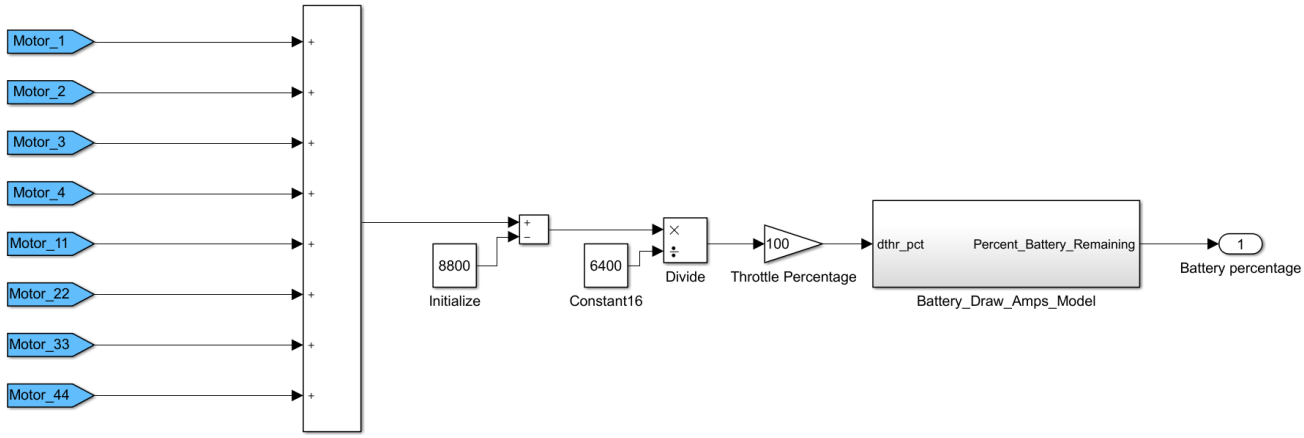

Figure 27 General architecture of the battery estimation model implemented in Simulink

Several flight tests were performed to evaluate the accuracy of the remaining battery estimation, the type of flight test conducted include pilot-in-the-loop flights and autonomous flights. Figure 28 shows one of the first results for the estimation model. In this case, the discharge rate was too slow compared to the actual battery discharge rate. To provide more precision to the model, the amperage draw of the motors was used as a tuning parameter. Is also important to note that external uncertainties such as wind will also affect the performance of the estimation model. For instance, if the day of flight test the wind is 6 mph the quadcopter will need higher thrust to maintain stability. Consequently, the battery will drain faster than a flight test with no wind.

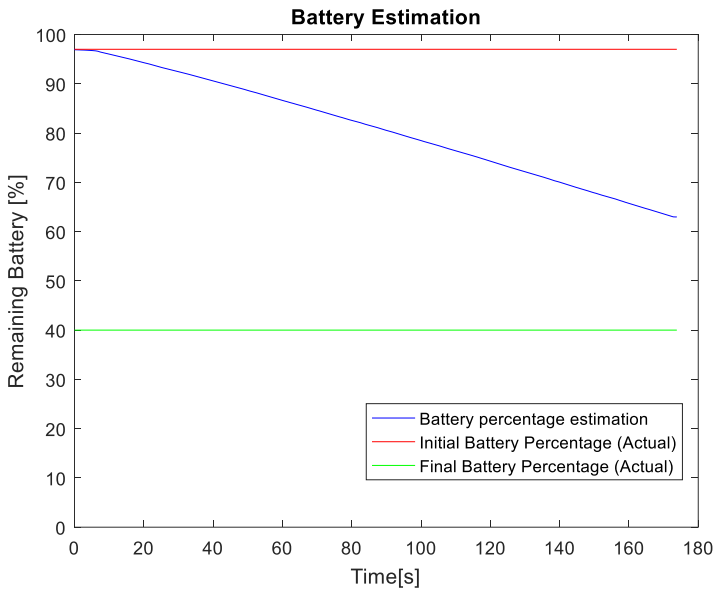

Figure 28 First test for the battery estimation model

Figure 29 shows the estimation results after giving the model a faster discharge rate; however, in this case, the discharge rate of the model was too fast compared to the actual discharge rate of the battery.

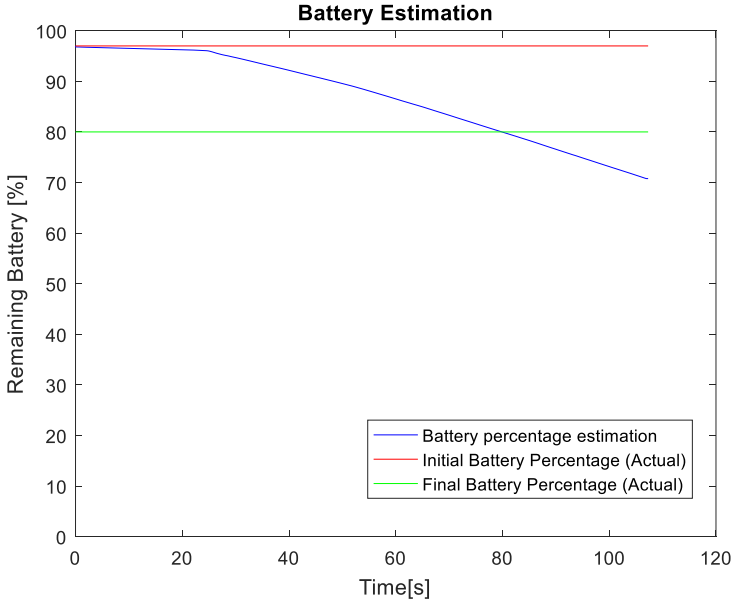

Figure 29 Results for battery estimation model after tuning

Therefore, further tuning was necessary to achieve better results. Figure 30 through Figure 32 confirm that the estimation quality increases with the tuning of the amperage draw of the motors.

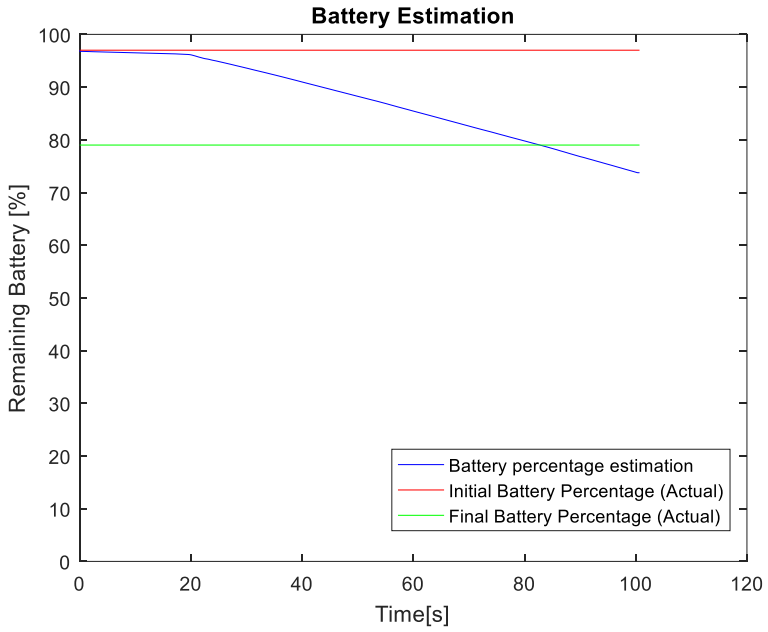

Figure 30 Battery estimation model

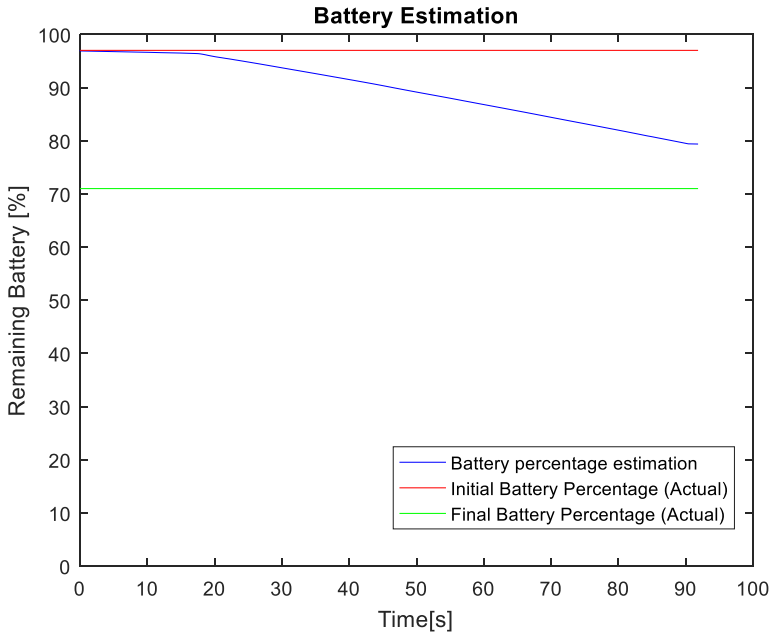

Figure 31 Battery estimation model with better performance

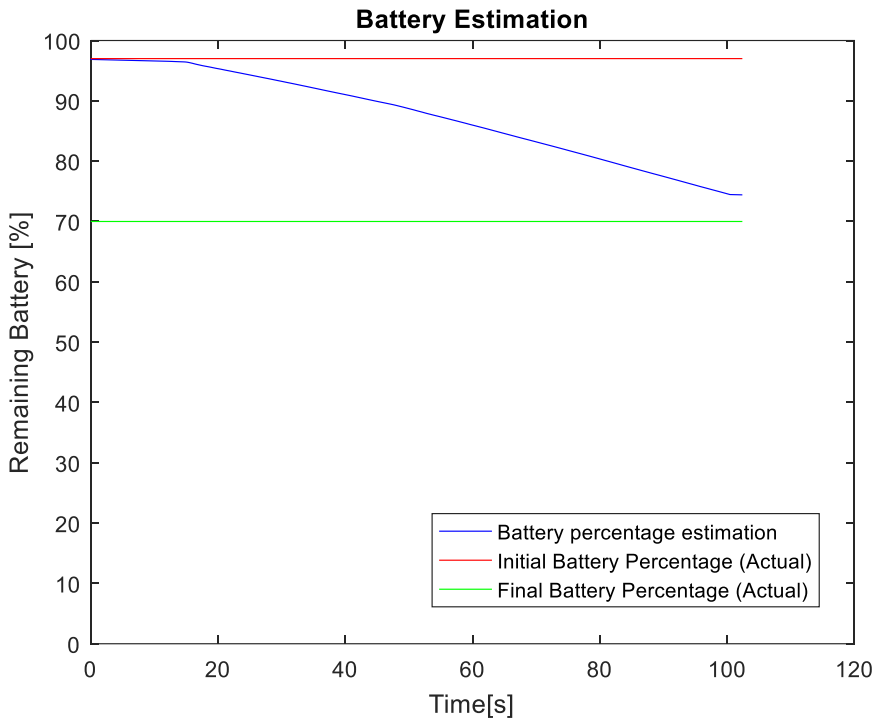

Figure 32 Battery estimation model with higher accuracy

With the battery information available, a decision to “return-to-home” or “in-situ landing” is taken in case of a low battery condition. It is important to mention that the path planner is used for the “return-to-home” maneuver, therefore it will take into consideration any obstacle the vehicle has already avoided to generate the RTH trajectory.

6. Simulation

In order to evaluate the robustness and viability of the proposed intelligent algorithms for autonomous decision-making, a simulation environment has been developed. MATLAB Simulink is used for the design of this simulation environment due to the resources and tools MATLAB provides to create environments that resemble real missions.

In the following subsections, the simulation architecture will be described as well as how all subsystems interact. The main subsystems in the simulation environment are sensors model, control laws model, quadcopter model, flight gear interface and intelligent algorithms model. In addition, simulation results for missions under abnormal conditions (such as unknown obstacle, failure in one motor and low battery percentage) are shown in this section. Is important to note, the scenarios shown in this section have been designed specifically to demonstrate the algorithm capabilities under a certain abnormal condition.

6.1. Quadrotor Simulation Environment

Figure 33 shows the general architecture for the 6DOF quadcopter simulation environment. This simulation integrates systems such as sensors data simulation, control laws for trajectory and attitude tracking, dynamic model of the quadcopter, and the intelligent algorithm scheme. Therefore, in the 6DOF simulation, the sensor model block is used to simulate navigation sensors such as the IMU (Inertial Measurement Unit) and GPS (Global Positioning System). The control laws model simulates the Nonlinear Dynamic Inversion (NLDI) baseline controller with an inner controller for the stability of the system and an outer controller for autonomous navigation (Garcia, 2017). In addition, the proposed intelligent algorithm is implemented within the control laws block. The quadcopter model represents the dynamics of the quadcopter, and a FlightGear Interface is

used to display the quadcopter performance during mission execution. These capabilities give the user a better understanding of the system performance under different circumstances.

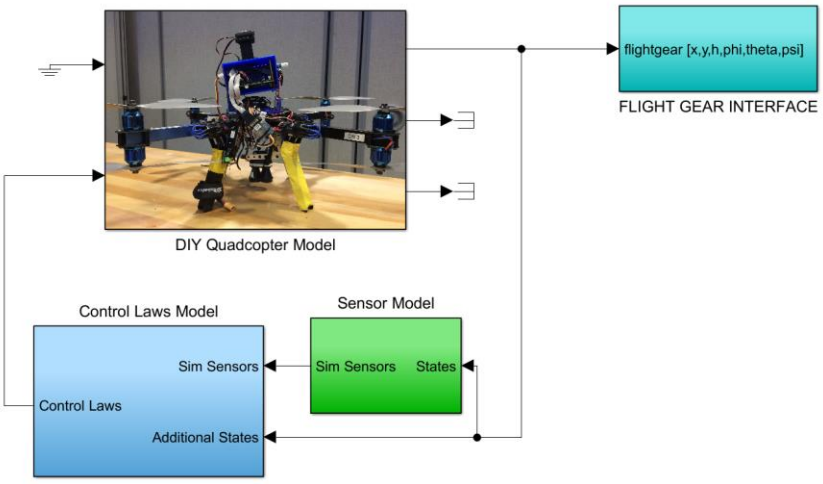

Figure 33 General architecture for the simulation environment

6.2. Control Laws

The main goal of the NLDI control approach is to transform a nonlinear system into a linear system through mathematical input-output feedback linearization. For the development of this type of controller, it is essential to derive with precision the equations of motion that describe the dynamics of the 6DOF quadcopter model, since the performance of the NLDI is highly dependent on it (Garcia, 2017). This approach is used for the tracking of desired outputs such as x, y, z positions, roll ($\emptyset$), pitch (θ), yaw (φ) angles, and p, q, r angular rates (Das, 2008, Ireland, 2015). Since the trajectory tracking dynamics are slower than the attitude dynamics of the vehicle, the controller is divided in two stages that are known as the outer loop and inner loop controllers (Moncayo, 2012). The general architecture of the NLDI implemented is presented in Figure 34.

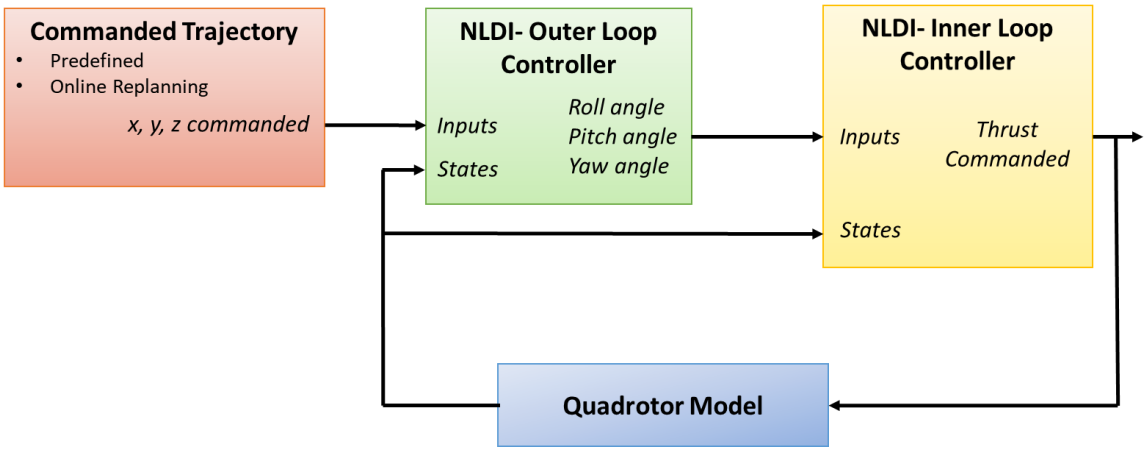

Figure 34 NLDI controller architecture

The commanded trajectory could be a predefined trajectory, or an online trajectory generated using the potential field path planner. For both cases, the controller receives x, y and z positions. These positions are the inputs for the outer loop controller. Using this information, two PID pseudo controllers were designed to obtain the desired attitude angles (roll, pitch, and yaw). The first pseudo controller calculates the desired velocities from the difference between the commanded and actual positions as shown in Eq.28 through Eq.30.

$$V_x = K_p(x_c - x_{actual}) + \int K_i(x_c - x_{actual})\,dt + \frac{d(K_d(x_c - x_{actual}))}{dt} \quad (30)$$

$$V_y = K_p(y_c - y_{actual}) + \int K_i(y_c - y_{actual})\,dt + \frac{d(K_d(y_c - y_{actual}))}{dt} \quad (31)$$

$$V_z = K_p(z_c - z_{actual}) + \int K_i(z_c - z_{actual})\,dt + \frac{d(K_d(z_c - z_{actual}))}{dt} \quad (32)$$

The direction cosine matrix (DCM), shown in Eq.31, is used to transform the velocities from earth to body reference frame. Eq.32 shows the equations for the reference transformation.

$$DCM_B^E = \begin{bmatrix} cos\theta cos\varphi & sin\emptyset sin\theta cos\varphi - cos\emptyset sin\varphi & cos\emptyset sin\theta cos\varphi + sin\emptyset sin\varphi \\ sin\theta sin\varphi & sin\emptyset sin\theta sin\varphi + cos\emptyset cos\varphi & cos\emptyset sin\theta sin\varphi - sin\emptyset cos\varphi \\ -sin\theta & sin\emptyset cos\theta & cos\emptyset cos\theta \end{bmatrix} \quad (33)$$

$$\begin{bmatrix} V_x \\ V_y \\ V_z \end{bmatrix}_B = DCM_B^E \begin{bmatrix} V_x \\ V_y \\ V_z \end{bmatrix}_E \quad (34)$$

With the velocities in the body reference frame, a second pseudo controller is designed to calculate the attitude angles using similar operations to the first pseudo controller.

$$\theta = K_p\left(V_{x\_c} - V_{x\_actual}\right) + \int K_i\left(V_{x\_c} - V_{x\_actual}\right) dt + \frac{d\left(K_d\left(V_{x\_c} - V_{x\_actual}\right)\right)}{dt} \quad (353)$$

$$\emptyset = K_p\left(V_{y\_c} - V_{y\_actual}\right) + \int K_i\left(V_{y\_c} - V_{y\_actual}\right) dt + \frac{d\left(K_d\left(V_{y\_c} - V_{y\_actual}\right)\right)}{dt} \quad (36)$$

Therefore, the outputs for the outer loop controller include the commanded pitch, roll, and yaw angles. The yaw angle is obtained through a different mathematical process. Subsequently, the attitude angles are the inputs for the inner loop controller, along with some output feedback states, which are required for the dynamic inversion process.

The inner controller is divided into two modes, the slow mode and fast mode. The slow mode concerns the dynamic inversion of the kinematic equations of the quadcopter to obtain the angular rates (p, q, r) from the desired attitude angles, shown in Eq.35 (Moncayo, 2012). The rates $\dot{\emptyset}, \dot{\theta}, \dot{\varphi}$ are calculated using the desired attitudes angles within a pseudo controller.

$$\begin{bmatrix} \dot{\emptyset} \\ \dot{\theta} \\ \dot{\varphi} \end{bmatrix} = \begin{bmatrix} 1 & \sin(\emptyset)\tan(\theta) & \cos(\theta)\tan(\theta) \\ 0 & \cos(\emptyset) & -\sin(\emptyset) \\ 0 & \sin(\emptyset)\sec(\theta) & \cos(\emptyset)\sec(\theta) \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix} \quad (37)$$

The fast mode implements the dynamic inversion of the moment equations of the quadcopter model, shown in Eq.36, to obtain the desired moments about each axis, where $\dot{p}, \dot{q}, \dot{r}$ are calculated using the desired angular rates from the inversion of Eq.35 and a pseudo controller. In addition, I_{xx}, I_{yy}, I_{zz} represent symmetrical moments of inertia of the quadrotor.

$$\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = -\begin{bmatrix} qr(I_{zz} - I_{yy})/I_{xx} \\ qr(I_{xx} - I_{zz})/I_{yy} \\ qr(I_{yy} - I_{xx})/I_{zz} \end{bmatrix} + \begin{bmatrix} 1/I_{xx} & 0 & 0 \\ 0 & 1/I_{yy} & 0 \\ 0 & 0 & 1/I_{zz} \end{bmatrix} \begin{bmatrix} M_x \\ M_y \\ M_z \end{bmatrix} \quad (38)$$

Therefore, the overall outputs for the inner loop controller are M_x the total rolling moment, M_y the total pitching moment, and M_z the total yawing moment. These moments are transformed to desired thrust, which is fed to the quadcopter plant closing the model loop. In this way, trajectory and attitude tracking of a quadcopter vehicle is achieved.

6.3. Intelligent Algorithm Simulation Integration

The intelligent algorithm model shown in Figure 35 is implemented along with the NLDI controller in the control laws model. This model includes the decision-making logic, the potential field path planner algorithm used for re-planning and trajectory generation, and the three modules for mission protection (Obstacle Detection, Health Monitoring, and Battery Estimation). The path planner algorithm is implemented using MATLAB Stateflow/Simulink. Stateflow is a MATLAB control logic tool that allows the user to design complex structures within a Simulink model. Stateflow is an advantageous tool that dynamically simulates switching and states changes within Simulink (Colgren, 2007). The Stateflow potential field architecture is shown in Figure 36.

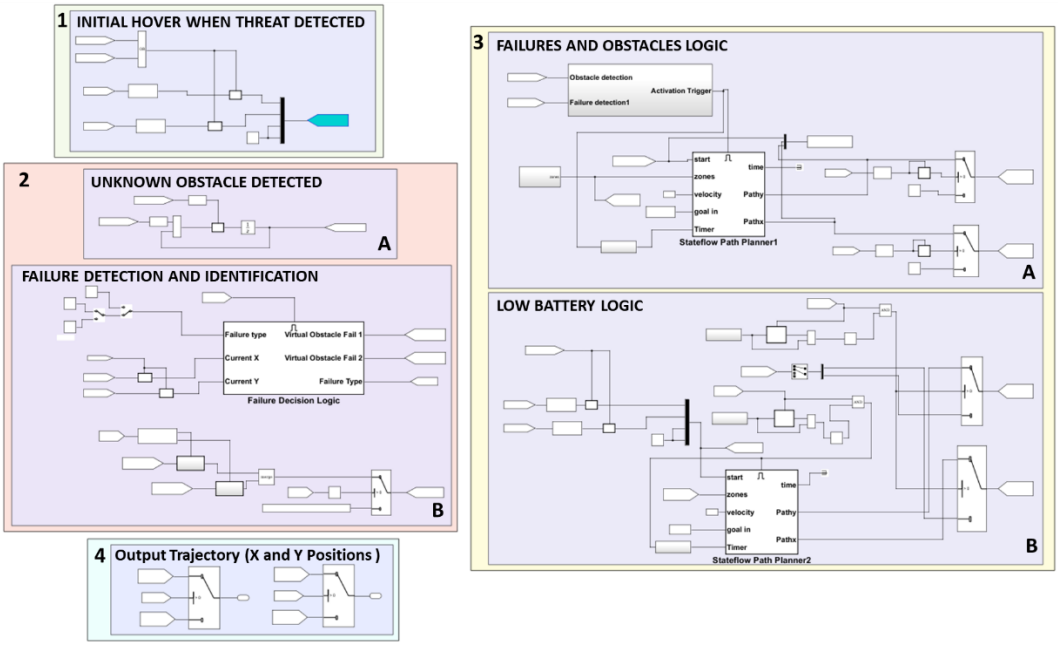

Figure 35 Intelligent Algorithm Model

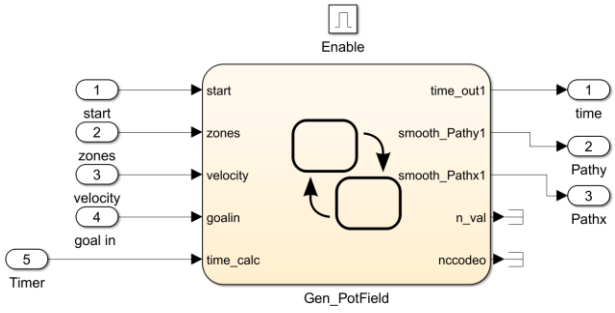

Figure 36 Stateflow Path Planner

A logic to simulate the information from the three decision-making modules was developed by implementing Simulink manual switches as shown in Figure 37. Detection of an unknown obstacle is simulated in the vision system using a comparison block. If the input value is greater than a certain value, an obstacle is detected. The detection of a failure or low battery percentage are simulated in a similar approach by comparing certain variables to a specific condition.

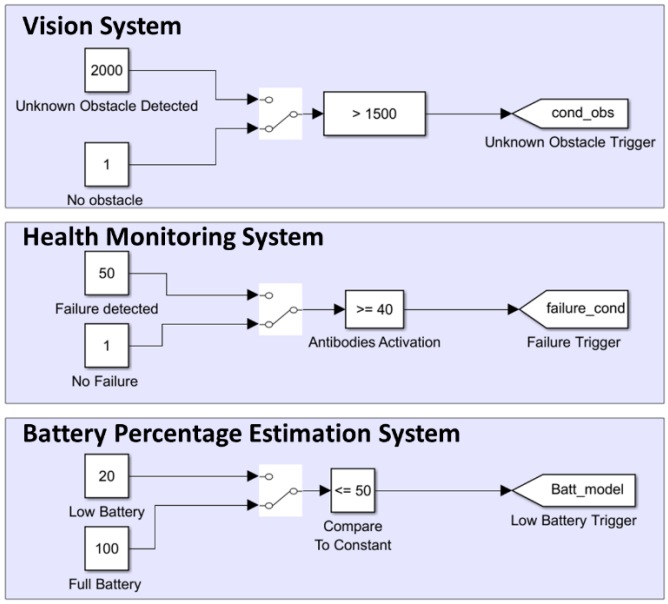

Figure 37 Simulation of decision-making subsystems

If any of the decision-making modules detects a threat, the intelligent algorithm receives the information and the decision-making process takes place. Each step shown in Figure 35 is described below for further understanding.

1. *"Initial Hover when Threat Detected"*

As aforementioned, a threat is considered as an unknown obstacle, a failure in the system, or low battery percentage. After a threat is detected, this step makes the vehicle hover for a few seconds to ensure the intelligent algorithm has sufficient time to calculate and generate the new mission tasks.

2. *"Unknown Obstacle Detected" & "Failure Detection and Identification"*

While the vehicle is in a hovering maneuver, the information received from the decision-making subsystems is processed in this step. Depending on the threat, a different action will be taken.

For unknown obstacles, the vision system detects and identifies the magnitude and

position of the obstacle encountered. The information is processed in step 2A, generating a trigger signal that will activate the path planner for obstacle avoidance. In addition to the trigger signal, the path planner receives the position and magnitude of the unknown obstacle.

For failure detection and identification, the health monitoring system detects an internal failure in the vehicle and identifies the type of failure. Once the information is received, the information is processed through the logic shown in Figure 35 and generates a virtual obstacle that makes the path planner generate a certain trajectory that benefits the vehicle performance the most, depending on the type of failure. The generation of the virtual obstacle is based on data obtained from training the system offline. Through the analysis of the vehicle performance under failures in simulation, the type of trajectories that benefit the system under different circumstances were identified.

The set of simulations conducted to understand what trajectory type to follow after recovering from a certain failure was designed using the same simulation environment as shown in Figure 33. For this research, two type of failures are considered. The first type of failure is a saturation in motor 1 and the second type of failure is a saturation in motor 2. Both motors are identified in Figure 38. The forward direction of the vehicle is indicated by the white symbol in the center.

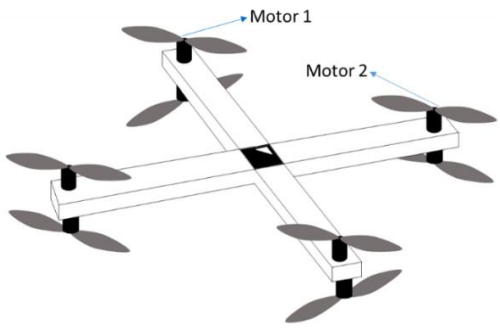

Figure 38 Motors subject to failure injection in the 3DR Quadcopter

Using the performance metrics previously discussed, the performance of the vehicle for two types of trajectories under two different abnormal conditions is analyzed. The trajectories generated shown in Figure 39 use the potential field algorithm to avoid the same obstacle at the position $\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 45.93 \text{ ft} \\ 0 \quad \text{ft} \end{bmatrix}$; the start and goal positions are the same for both cases.

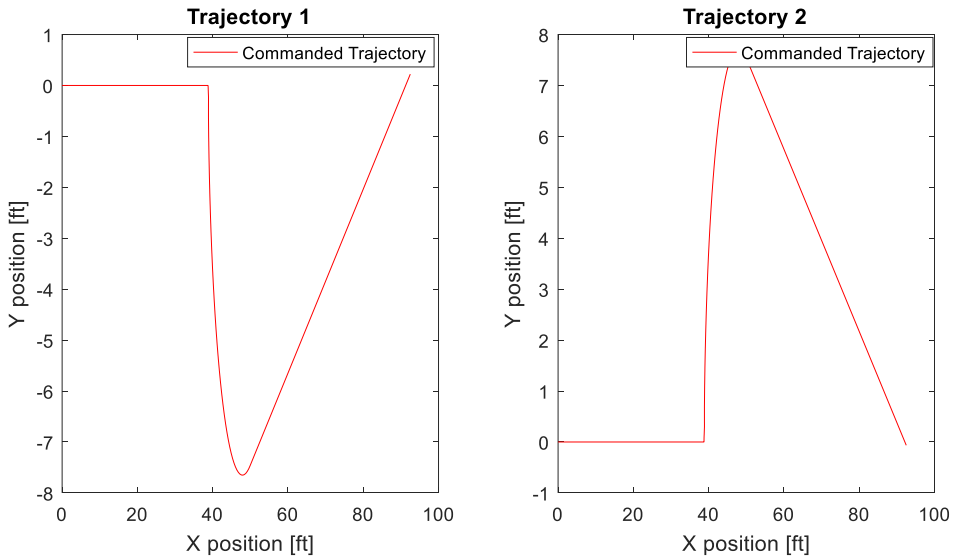

Figure 39 Commanded trajectories for autonomous navigation

Figure 40 demonstrates the response of the vehicle for each trajectory under nominal conditions. As mentioned before, performance metrics are implemented in the simulation environment to quantify the quality of the system's performance. For both trajectories, the Global PI is shown in Table 4. Table 4 shows that the response for both trajectories is good because both indexes are very close to each other and close to 1. This means the error between the commanded and actual states is close to zero.

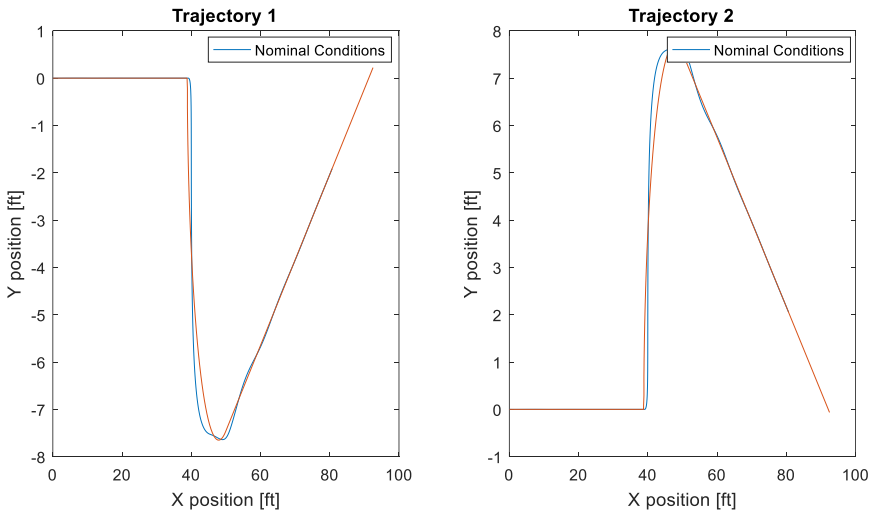

Figure 40 Vehicle performance under nominal conditions

Table 4 Vehicle Global PI for Trajectory 1 and Trajectory 2 nominal conditions

Quadrotor simulation under nominal conditions	
Trajectories generated with PF	*Global PI for vehicle performance*
Trajectory 1	0.7842
Trajectory 2	0.7816

Failure in motor 1, simulated as a saturation, is the first unexpected event to test the vehicle response. Figure 41 exhibits the vehicle behavior under this type of failure, which shows that Trajectory 1 is more favorable than Trajectory 2 as the control effort to track the desired trajectory is less. The calculated PI for both trajectories as displayed in Table 5 validates this behavior.

Table 5 Vehicle Global PI for Trajectory 1 and Trajectory 2 for failure in motor 1

Quadrotor simulation under abnormal conditions	
Trajectories generated with PF	*Global PI for vehicle performance*
Trajectory 1	0.5431
Trajectory 2	0.4976

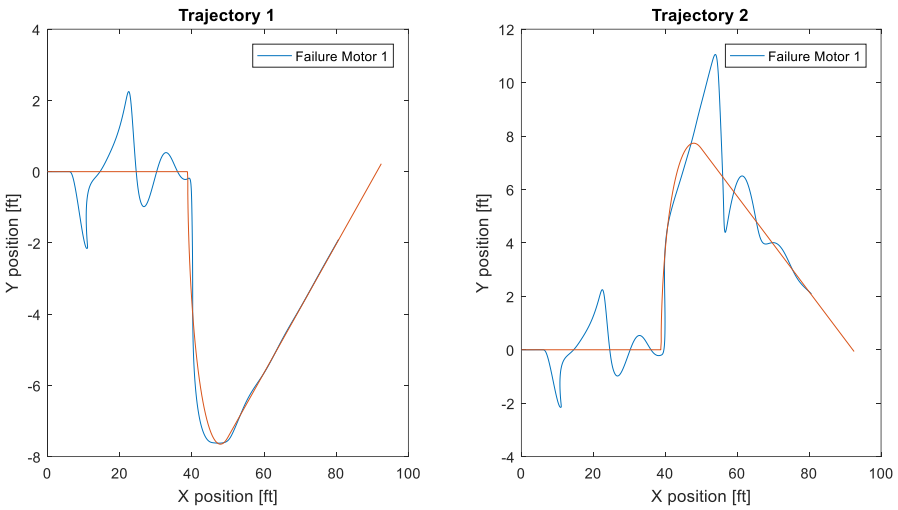

Figure 41 Vehicle performance under failure condition in motor 1

Assuming a failure is detected by the HM and is properly identified as a failure in motor 1, the system will make the decision to generate trajectories similar to Trajectory 1 to enhance mission safety. Similarly, Figure 42 shows the behavior under failure in motor 2 for both trajectories. The results for the Global PI are also presented in Table 6. From the results obtained it is clear that Trajectory 2 is more favorable for a failure detected in motor 2. Therefore, the system under a failure in motor 2 will choose trajectories similar to Trajectory 2 to assure safe and efficient navigation.

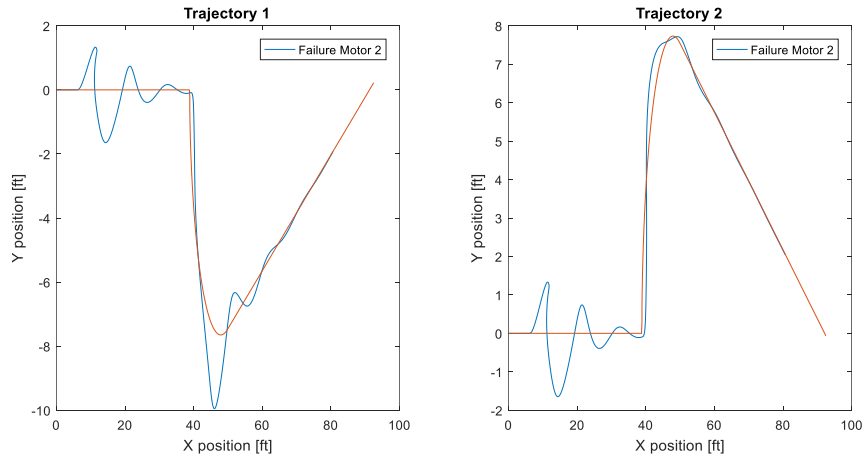

Figure 42 Vehicle performance under failure condition in motor 2

Table 6 Vehicle Global PI for Trajectory 1 and Trajectory 2 for failure in motor 2

Quadrotor simulation under abnormal conditions	
Trajectories generated with PF	*Global PI for vehicle performance*
Trajectory 1	0.5482
Trajectory 2	0.5811

A database is generated using the previous results, and the system is trained to perform intelligent decision-making without human supervision should any of these failures be detected while performing the mission. Therefore, using this knowledge the system will generate a virtual obstacle in step 2B, shown in Figure 35, to ensure the generation of the most favorable trajectory for the mission. It is important to mention that as the trajectories generated are calculated in the Earth reference frame the system identifies in which quadrant in the coordinate plane the quadcopter is flying. Knowing the quadrant is important for the position of the virtual obstacle. For example, if the quadcopter is following Trajectory 1, and there is a failure in motor 1, Figure 43 shows how the position of the virtual obstacle depends on the quadrant in order to ensure that the replanned trajectory benefits the performance of the vehicle, which will be a trajectory similar to Trajectory 2.

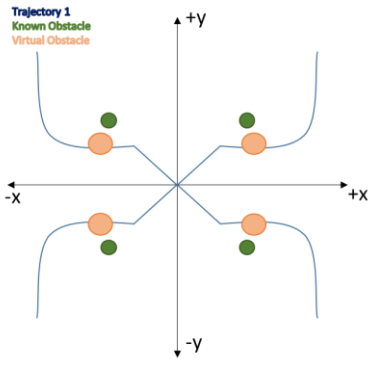

Figure 43 Virtual obstacle dependence on coordinate plane quadrants

After the virtual obstacle's position and magnitude have been defined, the information is sent to the path planner along with a trigger command for the path planner activation.

3. *"Failures and Obstacles Logic" & "Low Battery Logic"*

Step 3 is where the trajectory generation takes place; the path planer takes all the information received from step 2 and calculates the optimal trajectory to complete the mission. Due to logic complexity, one path planner block is used for two threat cases that are a failure in the system and an unknown obstacle in the path. Another path planner block, as shown in Figure 35, is used for the threat in the case of low battery in the system. It is important to mention that the two path planner blocks are the same; the difference remains on the inputs each block receives.

4. *"Output Trajectory"*

As a final step to complete the decision-making process, the path planner outputs the optimal calculated trajectory in X and Y coordinates. These coordinates are taken with respect to the initial starting point of the quadcopter trajectory. The desired altitude is fixed for the simulation. This information is the input to the NLDI outer-loop controller as the desired position to complete the mission. If any threat is detected again, the intelligent system takes action and the decision-making process repeats itself.

6.4. Performance Evaluation Case Study

The response of the presented intelligent algorithm was tested for each of the three threats considered in this thesis: unknown obstacles, failures in the system, and low battery percentage.

6.4.1. Unknown Obstacle

In Figure 44, one known obstacle displayed as green is provided to generate an original path, which the vehicle is commanded to follow. In addition, two unknown obstacles, shown as red, are randomly added after the mission starts.

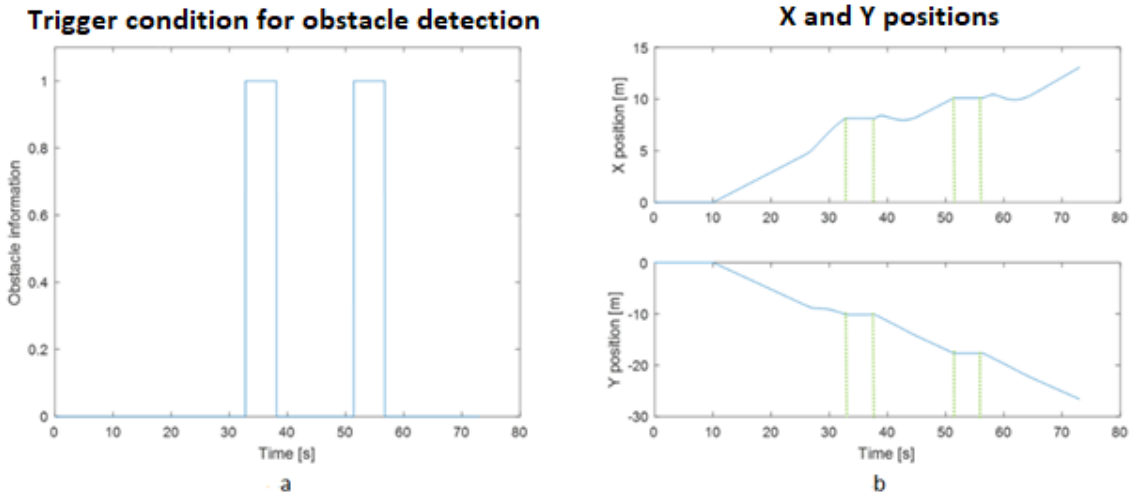

Figure 45a shows the trigger condition provided by the vision system, which is activated once an obstacle is detected. From this activation status, it can be shown in

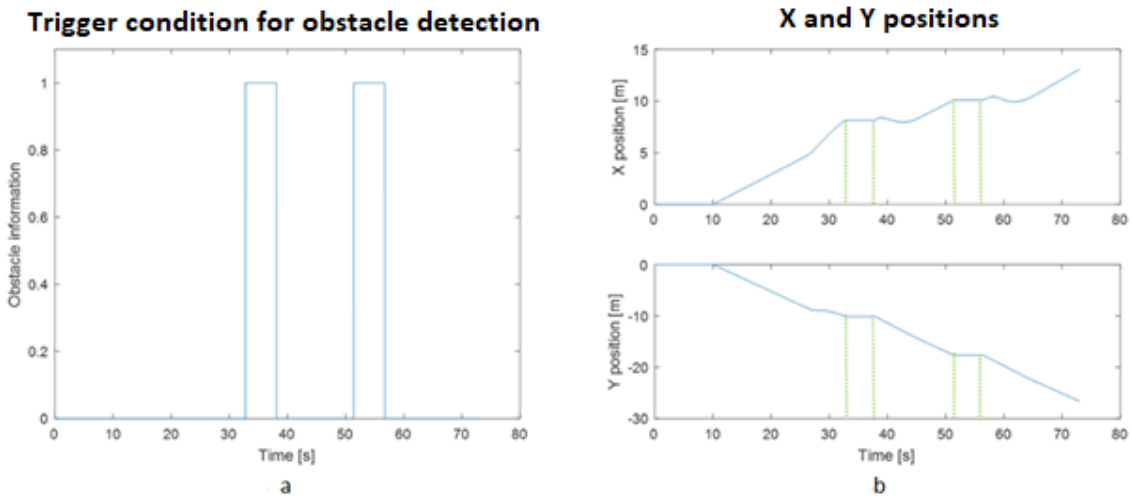

Figure 45b that the vehicle is holding position while the path planner provides a new trajectory to successfully avoid the unknown threat. It is important to note that the detected threat can represent not only the obstacles but also radar or other restricted areas

that the vehicle would be unauthorized to cross.

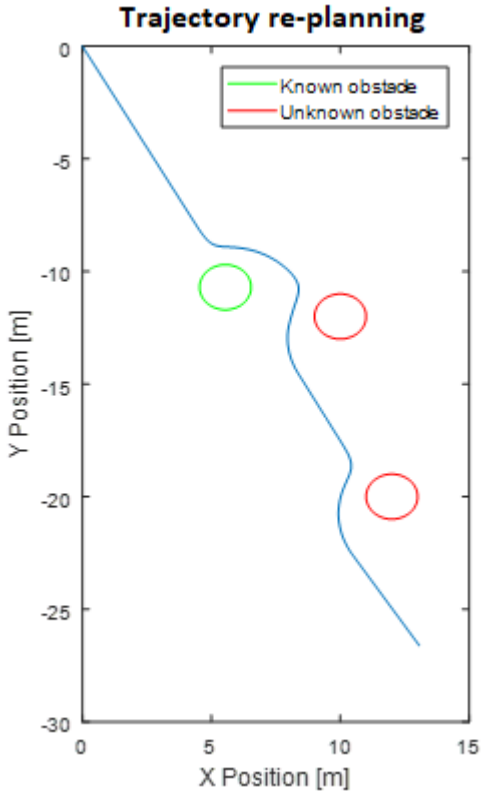

Figure 44 Re-planning capabilities due to unknown obstacles

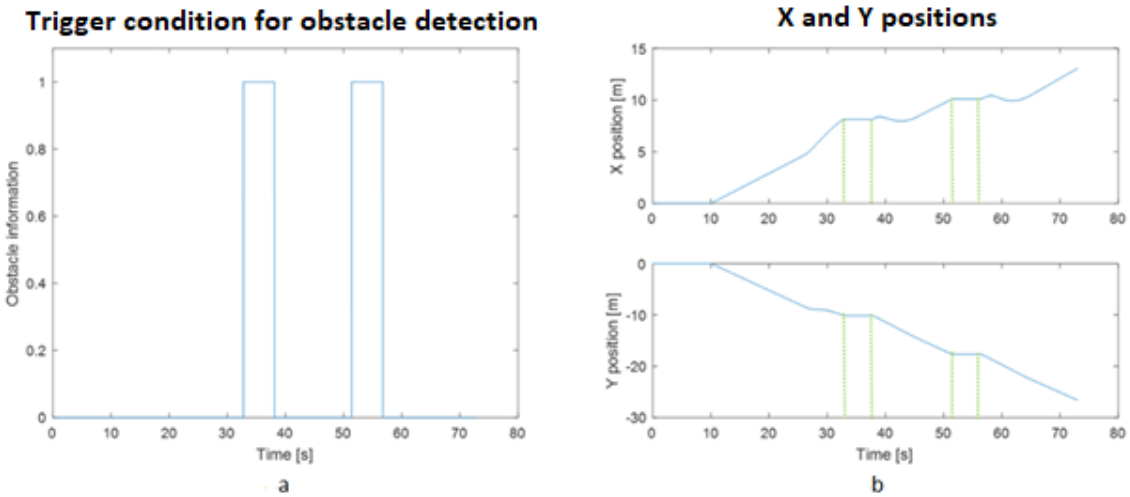

Figure 45 Trigger activation for obstacles detection (left), and commanded x and y trajectory (right)

Figure 46 shows the quadcopter response obtained for autonomous re-planning due to unknown obstacles. In this example, the intelligent algorithm is implemented within the

simulation environment shown in Figure 33. Is important to mention that the unknown obstacles position is based on the current position of the quadcopter. In other words, the current position of the vehicle at the failure time t_f is taken and by adding some magnitude in X and Y, the position of the unknown obstacle is simulated. The green circles in Figure 46 represent the unknown obstacles detected by the system, the black circle is the known obstacle, the red path is the commanded trajectory, and the blue path is the actual response from the vehicle.

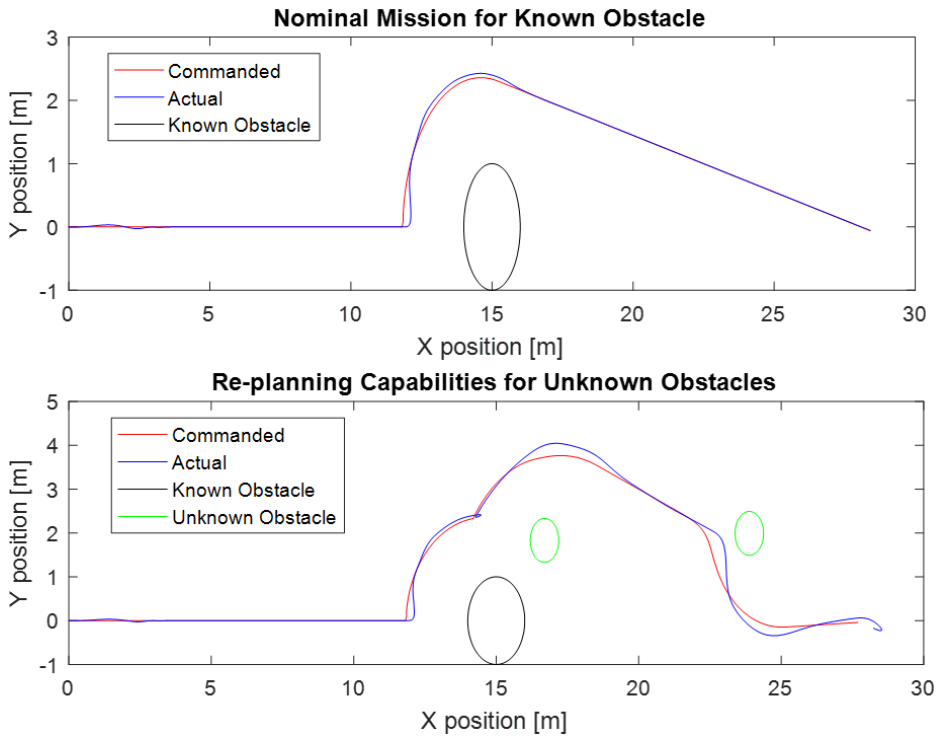

Figure 46 Vehicle autonomous re-planning for unknown obstacles in the mission

6.4.2. System Failures

Figure 47 shows two simulation results from the intelligent algorithm alone for two different failures, failure motor 1 and motor 2. As shown in the figure, after a failure is detected and identified by the HM, a virtual obstacle is created through the decision-making process and a new trajectory is generated. In both cases, the new trajectory benefits

the performance of the vehicle based on knowledge acquired from the analysis conducted in the previous section.

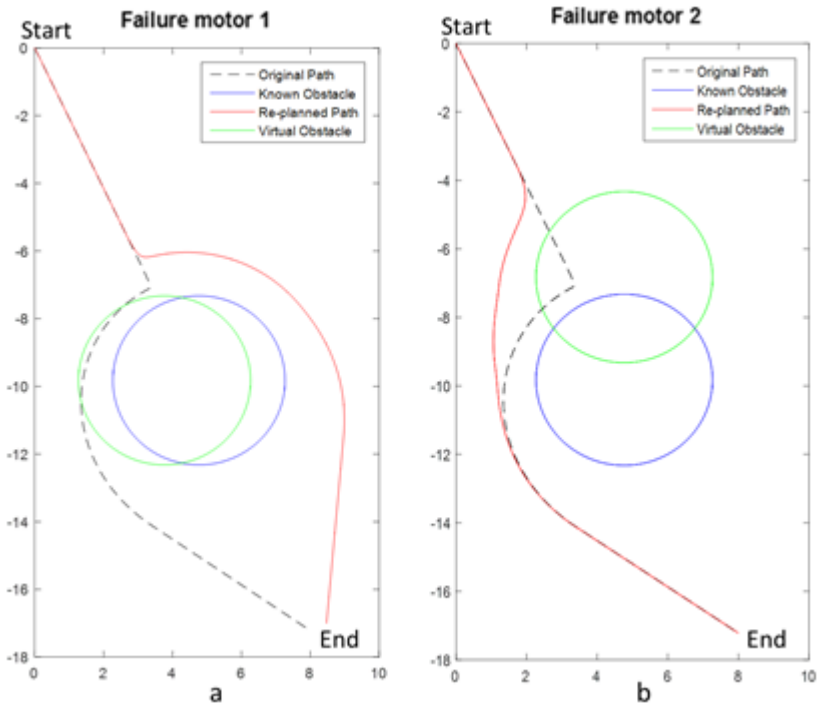

Figure 47 Intelligent algorithm for failures in the system

Figure 48 presents another example to demonstrate decision-making and re-planning capabilities using the same start, goal and obstacles positions as the example in Figure 45. The only known obstacle is shown in green and the red obstacles are unknown obstacles detected by the vision system. Similarly, as in the obstacle avoidance case, in Figure 48 the vehicle is avoiding the known and unknown obstacles; however, in both cases, the system presents a failure in motor 1 (Figure 48a) and a failure in motor 2 (Figure 48b). The failure information is obtained from the HM, which detects and properly identifies the failure. The offline training aforementioned is used to generate the trajectories that are more favorable for the vehicle performance. Therefore, as seen in Figure 48a, due to failure in motor 1 the trajectory is similar to Trajectory 1, and the trajectory in Figure

48b is similar to Trajectory 2 due to a failure in motor 2.

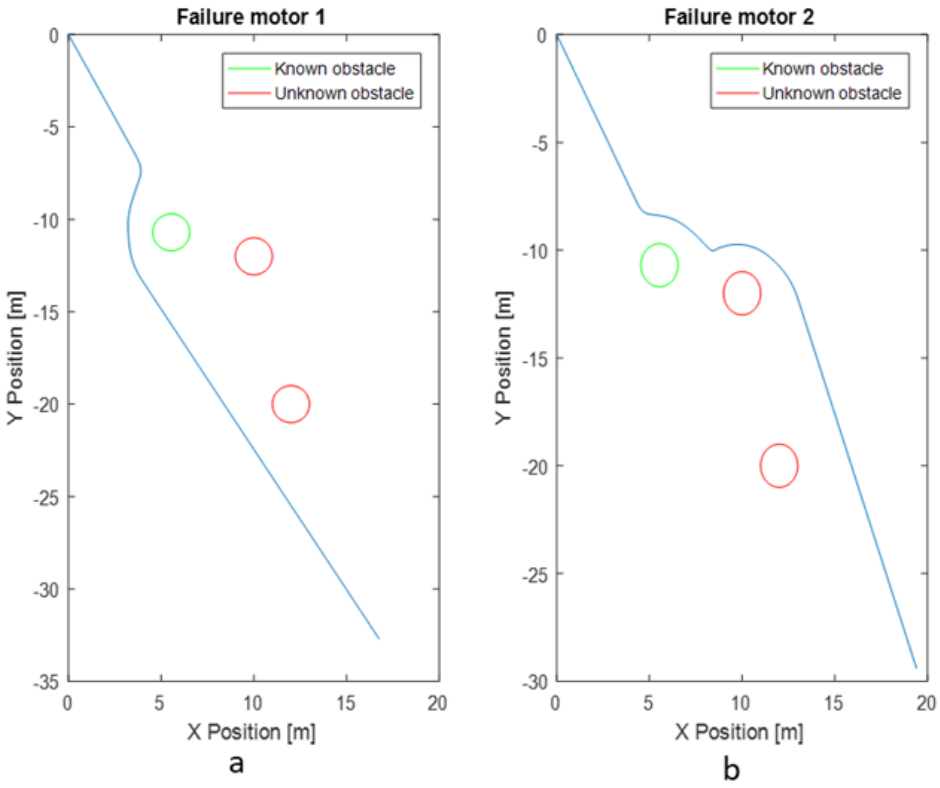

Figure 48 Decision-making and re-planning capabilities under abnormal conditions

Figure 49 and Figure 50 show the quadcopter response obtained for autonomous re-planning due to failures in the system. In both results, the blue path represents the actual response from the quadrotor, the red path represents the commanded trajectory, the black circle in a known obstacle, and the green circle represents the virtual obstacle generated by the intelligent algorithm to ensure the vehicle will complete the mission safely. For the results in Figure 49, the vehicle was subject to a failure in motor 1 while tracking a trajectory similar to Trajectory 2 in Figure 40. Based on preliminary analysis, it is known that the quadcopter under failure in motor 1 will have a better performance by re-planning a trajectory similar to Trajectory 1 shown in Figure 40. The

results for re-planning shown in Figure 49 confirm this conclusion.

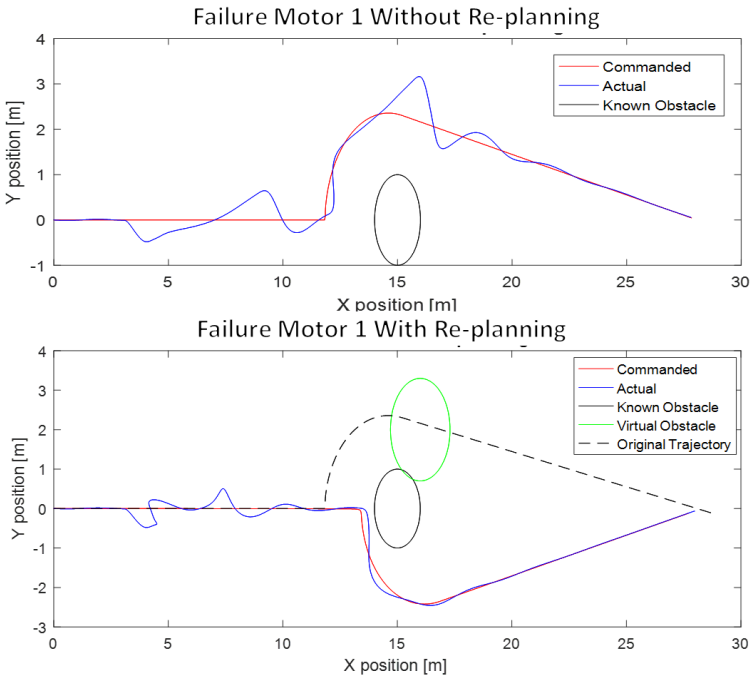

Figure 49 Re-planning capabilities due to failure in motor 1

Similarly, the quadcopter was subject to a failure in motor 2 while tracking a trajectory similar to Trajectory 1. The results shown in Figure 50 confirm that the vehicle's performance improved by re-planning a trajectory similar to Trajectory 2.

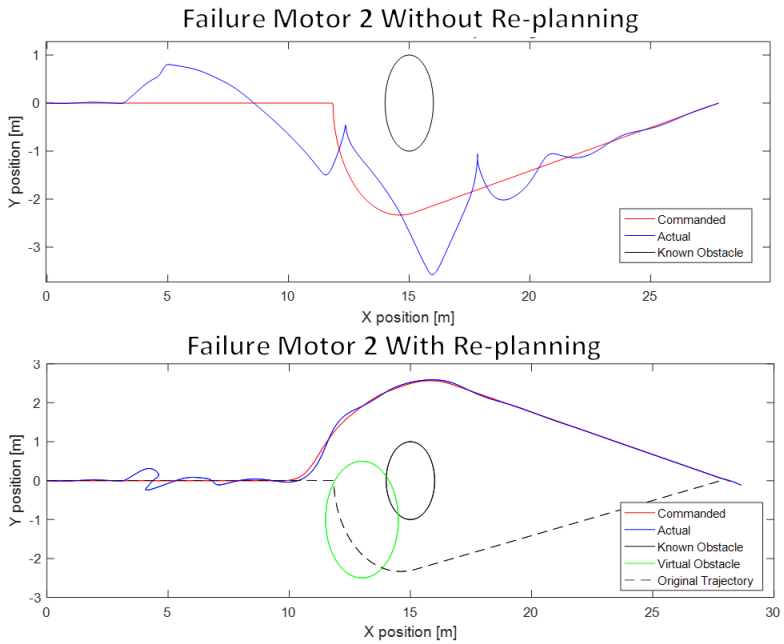

Figure 50 Re-planning capabilities due to failure motor 2

6.4.3. Low Battery Condition

As an example, a return-home capability was implemented in simulation, and it is shown in Figure 51. After low battery has been detected (Figure 51 b), the vehicle is commanded to return-to-home by following the proper x-y trajectories (Figure 51 a, and c). It is important to mention that even for returning home the obstacle in the trajectory is taken into account for re-planning (Figure 51d).

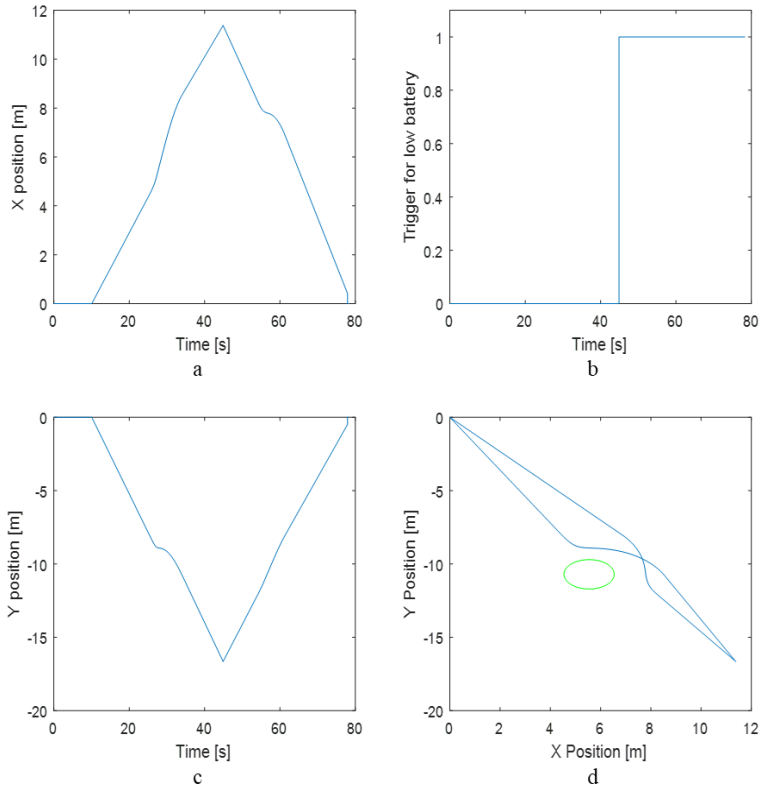

Figure 51 Return to home maneuver

In the same way as for the previous cases, the logic for low battery was implemented within the quadcopter simulation. Figure 52 shows the re-planning capabilities due to low battery. In this case, the quadcopter performs a return to home maneuver where the red path represents the commanded trajectory and the blue path represents the actual trajectory performed by the vehicle. Is important to note that the path planner considers the known obstacle for obstacle avoidance while returning home.

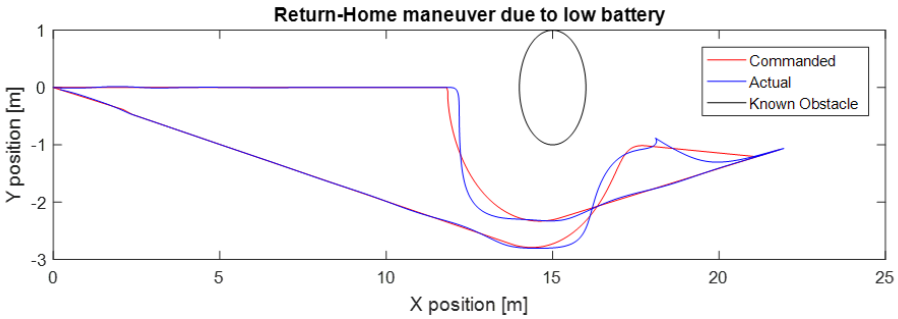

Figure 52 Re-planning capabilities due to low battery

7. Implementation

7.1. UAV Testbed & Hardware Tools

In order to test the algorithms designed in simulation, a 3DR X8 quadcopter testbed has been used for implementation and testing. The vehicle is shown in Figure 53.

To provide the quadcopter with the necessary capabilities to perform autonomous missions and pilot-in-the-loop maneuvers, the vehicle has been instrumented with several hardware components that are explained in this section.

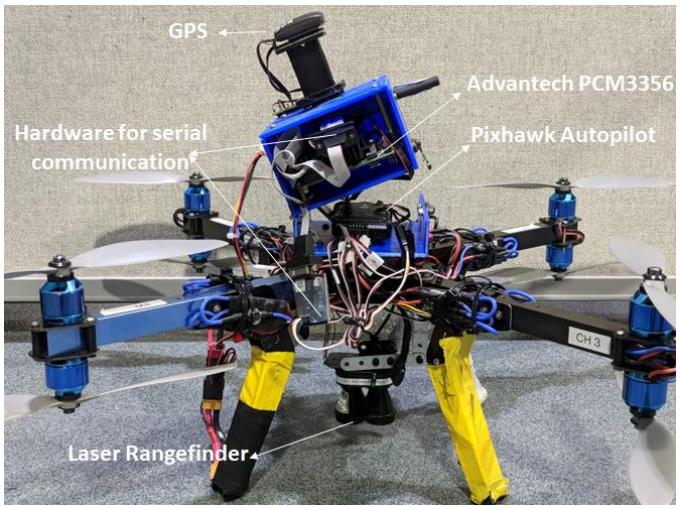

Figure 53 3DR X8 vehicle for algorithms testing

The main features of the quadcopter testbed are presented in the following table:

Table 7 Main features of the quadrotor testbed

Features	Dimensions
Total Flight Weight	2.99 kg
Thrust Arm	27.43 cm
Height	36.07 cm
Ixx	0.0132 $Kg * m^2$
Iyy	0.0132 $Kg * m^2$
Izz	0.023 $Kg * m^2$

7.1.1. Onboard Computers

To process all the information and integrate the navigation sensors, two onboard computers were selected. The first one is the Pixhawk 1 Autopilot shown in Figure 54. The Pixhawk was selected due to its low cost and its capabilities to operate in real time. Therefore, the Pixhawk runs all the control laws for stabilization and tracking along with the protocols to read and process sensors information. This autopilot module uses a 168 MHz Cortex M4F CPU, it has 14 PWM/servo outputs and abundant connectivity options for additional peripherals (UART, I2C, and CAN).

Figure 54 Pixhawk Autopilot onboard computer

The secondary onboard computer is the PC104 Advantech PCM-3356. The Advantech features an AMD G-Series T16R 615 MHz/T40E 1.0Ghz processor with onboard memory capabilities of 4GB DDR3L SODIMM/1G DDR3L. Advantech offers the capability of enabling Windows Embedded software products. Some of the drivers compatible with Advantech for installation include Windows XP Professional, Windows OS, and Linux. In addition, the Advantech platform has released a program library called Secured and Unified Smart Interface (SUSI) that will enable users to access hardware through drivers integrated into the SUSI library.

In this thesis, the PC104 initializes through an external DOS bootable USB drive.

The PC104 is chosen because of its enhanced processing and memory capabilities in addition to its real time processing capabilities. This onboard computer runs the intelligent algorithms, receives, and sends information to the Pixhawk. The Advantech computer is shown in Figure 55.

Figure 55 PC104 Advantech PCM-3356 onboard computer

7.1.2. Navigation Sensors

To obtain the information the controller and intelligent algorithm need, the following hardware was chosen to acquire data such as Euler angles, angular rates, position in the x, y and z coordinates, accelerations and more.

- Invensense MPU6000 3-axis Accelerometer/Gyroscope

  The MPU6000 is the primary accelerometer and gyroscope used by the Pixhawk and is shown in Figure 56.

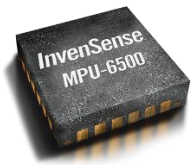

Figure 56 MPU6000 inertial sensor

- ST Micro L3GD20H 16 bit Gyroscope

  The L3GD20H 16 bit gyroscope is a secondary gyroscope sensor equipped in the Pixhawk for data redundancy. It is a low-power angular rate sensor which

gives the possibility to communicate the measured angular rates at different bandwidths to external devices through I^2C interface.

- ST Micro L3M303D 14 bit Accelerometer/Magnetometer

  The L3M303D 14 bit accelerometer/magnetometer is a secondary accelerometer/magnetometer sensor equipped in the Pixhawk. It includes an I^2C interface used to communicate readings through external connections.

- UBLOX LEA-6H GPS Receiver Module with Digital Compass

  The external UBlox GPS includes the HMC5883L digital compass which is connected to the Pixhawk. The sensor provided the position information to the Pixhawk onboard computer and is shown in Figure 57.

Figure 57 GPS receiver module with digital compass

- Lightware SF11/C Laser

  The Lightware SF11/C laser is connected to the Pixhawk and is used for altitude readings for autonomous missions. The laser is connected to the Pixhawk through serial. The Lightware SF11/C radar has a range up to 120m for distance readings of solid surfaces making it ideal for implementation in the 3DR X8 quadcopters. A micro USB port is available on the laser for configuration purposes. The Lightware SF11/C laser is shown in Figure 58.

Figure 58 Lightware SF11/C Laser

7.1.3. Communication Hardware

- DX8 8CH Transmitter with AR8000/TM1000 and DSMX Remote Receiver

The DX8 8CH Transmitter with AR8000 8 channel high-speed receiver shown in Figure 59 are used for pilot-in-the-loop commands and enabling autonomous missions through signals from the Transmitter. The signals from the DX8 8CH transmitter are received by the DSMX remote receiver connected to the Pixhawk.

Figure 59 DX8 8CH transmitter and DSMX remote receiver

- TTL to RS232 converter

To establish serial communication between the Pixhawk and the Advantech computer, two TTL to RS232 converters, shown in Figure 60, are used. This communication is essential since the Pixhawk sends the current states information to the Advantech computer for the intelligent and decision–making algorithms such

as health monitoring and battery percentage estimation. In addition, the Advantech computer sends to the Pixhawk the path information to follow in an autonomous mission with re-planning capabilities.

Figure 60 TTL to RS232 converter

7.1.4. Power and Propulsion Hardware

- 4S Lipo Battery

A 4 cells 14.8V 5000mAh LiPo Battery, shown in Figure 61, is used to supply power to all onboard systems during flight.

Figure 61 4S Lipo Battery

- UBEC DC/DC step-down converter

The Advantech computer needs to be powered with 5V. Therefore, a low-cost UBEC DC/DC step-down converter, shown in Figure 62, is used to convert the 14.8V from the Lipo Battery to 5V.

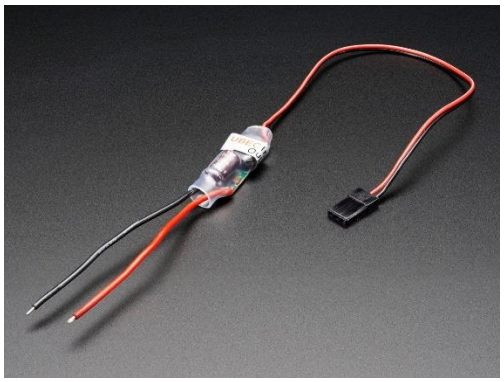

Figure 62 UBEC DC/DC step-down converter

- Motor 850 Kv AC 2830 – 358

To supply the required thrust for the vehicle, eight electric AC brushless motors are used. The motor shown in Figure 63 is rated for 850 RPM/V [Kv]. These low-cost and durable motors assure full authority throughout the flight envelope.

Figure 63 3DR X8 Motors

- 3D Robotics 20A Electronic Speed Controller (ESC)

To control the motor speed, 3D Robotics 20A ESCs with Simonk Firmware are used as shown in Figure 64.

Figure 64 20A ESC

- APC 10x4.7 SF Plastic Propellers

  Propellers with a diameter of 10 inches and a geometric pitch of 4.7 inches are the recommended size to be used with the AC brushless motors. Four of the propellers have the 'pusher' configuration to be compatible with the reverse rotation. Plastic propellers as shown in Figure 65 were chosen for their low-cost.

Figure 65 10x4.7 Propeller

7.2. Embedded Code Software Tools

For implementation purposes, the same architecture used in simulation was developed. An NLDI controller is used as a baseline controller for attitude and trajectory tracking. To increase the system's robustness, the NLDI controller has been augmented with an adaptive controller (Perez, 2016) to help the system reject unknown disturbances

such as wind, or help the system recover from unknown failures.

Figure 66 shows the general architecture of the code developed with MATLAB/Simulink for implementation of the control laws in the Pixhawk onboard computer. The difference from the simulation environment relies on the implementation of different protocols and Simulink block diagrams to enable the communication between the onboard computer and all the sensors previously mentioned. Most of the block diagrams in Simulink are supported by the Pixhawk support package. The intelligent algorithm shown in Figure 35 is implemented in the Advantech computer. Both codes (Pixhawk & Advantech) contain serial protocols that enable a two-way communication (send/receive) between the onboard computers. Key software tools will be described in the following sections to further understand the integration of all the communication protocols within the implementation code.

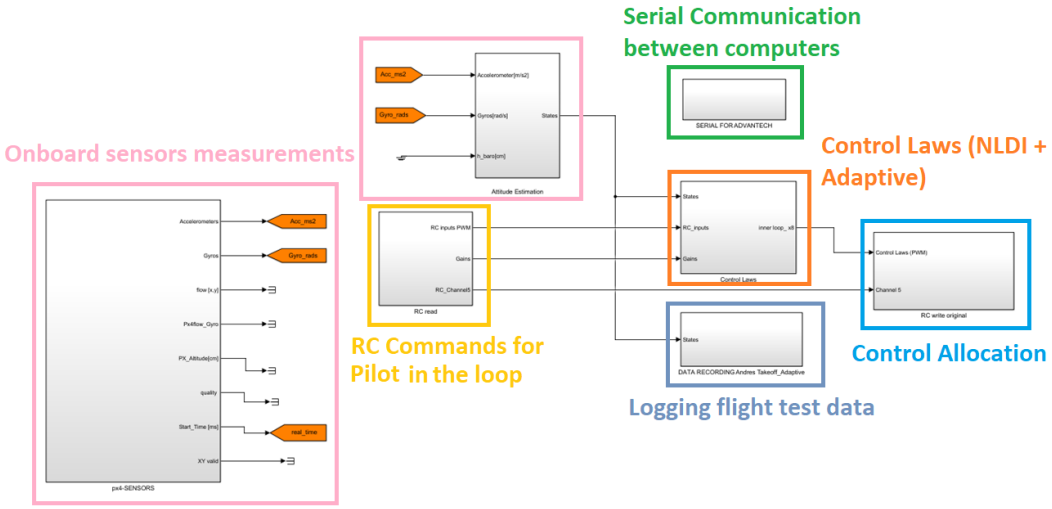

Figure 66 General Architecture for Implementation code developed with MATLAB/Simulink

7.2.1. S-Function Development Tools & Serial Protocol

MATLAB/Simulink S-Function blocks help users create an interface between simulation and hardware. The S-Function can be coded using MATLAB, C, C++, or Fortran. Figure 67 shows the S-Functions developed at the Advanced Dynamics and Control Lab. These tools enable the communication between the Pixhawk onboard computer and some of the sensors detailed in previous sections such as the GPS (PX4_gps_position), SF11/C laser, and gyroscopes (PX4_gyros_filtered). These blocks are integrated within the *"Onboard sensors measurements"* blocks shown in Figure 66. The GPS is applied more for autonomous missions, where is important for the controller to know the actual position of the vehicle. The laser main function is to measure the altitude during flight, and it is also mainly used for autonomous flights. Last but not least, the gyroscope block gives the values for the angular rates $p, q,$ and r.

The S-Function "Receiver" shown in Figure 67 is used to receive data from the Advantech computer through the TTL to RS232 converter. The S-Function is integrated within the *"Serial communication between computers"* block. This block also includes the "Serial UART" block, shown in Figure 67, which is supported by the Pixhawk support package and it is used to send data from the Pixhawk board to the Advantech computer using the TTL to RS232 serial converter.

In addition, an S-Function has been developed to log the sensors data from flight tests. This block (PX4_signal_log) is a key element for post-flight data analysis and is introduced in the *"Logging flight test data"* block shown in Figure 66.

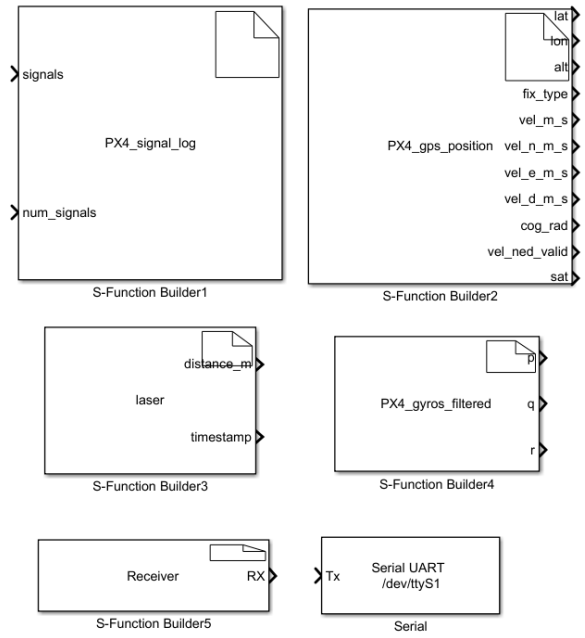

Figure 67 S-Function blocks for sensors data reading

7.2.2. Pixhawk Support Package Tools

The Pixhawk support package includes libraries that allow the user to interact with IMUs (Inertial Measurement Units), PWM (Pulse Width Modulation), RC Control, and more. Figure 68 shows some of the blocks that are used in the implementation code shown in Figure 66 (Pixhawk Support Package, 2017).

The “input_rc” is also a key element for the operation of the quadcopter; it allows the pilot to control the vehicle, to enable autonomous missions, to kill (i.e. cut power) the system if it becomes unstable, to name few functions. The “input_rc” is located inside the block *“RC Commands for pilot in the loop”* in Figure 66. The “PWM_output” sends to all the motors the required throttle commands in PWM signals, and it can be found inside the block *“Control Allocation”* in Figure 66. The “RDB_LED” is a LED light in the Pixhawk board to indicate three main status, red means the vehicle is killed (i.e power is not being sent to the motors), green means the vehicle is ready to fly, and blue means the vehicle is

ready but there are no GPS readings; this block is also within *"Control Allocation"*. An additional block inside *"Control Allocation"* is the "Speaker_Tune", which has been designed to make a buzz when the vehicle finds GPS readings, and to indicate in autonomous flights when the mission has started. The "sensors_combined" and the "vehicle_attitude" tools provide the accelerations and attitude measurements, which are used in the control laws explained heretofore. Both blocks are part of the *"Onboard sensors measurements"* sections.

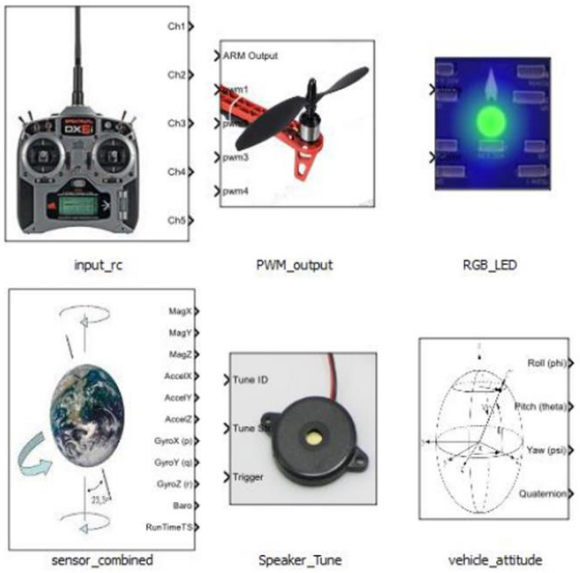

Figure 68 Pixhawk Support Package library

With all the software and hardware components discussed, the vehicle is equipped with several features to be a robust, reliable and intelligent system. To prove these capabilities several flight tests have been performed and the results are presented in the following section.

7.3. Intelligent Algorithm Evaluation Implementation

The aim of this section is to evaluate the outcome of the proposed algorithm in real-time implementation. First, it is important to describe the general process of planning, preparing, and executing a mission. The selected field to perform flight tests is shown in Figure 69a, which has been approved for UAV flight testing by Embry-Riddle Aeronautical University. Before any flight, a predefined trajectory is generated using either potential field or Clothoid algorithms. The start and end of the trajectory are selected from Mission Planner. Mission Planner is a ground control station for UAVs that allow users to plan autonomous missions, load firmware in the Pixhawk board, monitor the vehicle's status during operation, and more. For this thesis, Mission Planner is only used to select the initial and final trajectory points and the location of any known obstacle. Figure 69b shows an example of the field in Mission Planner and the points selected for the start, end, and known obstacles of the trajectory. With this information, any of the two path planners studied in this thesis can be used for trajectory generation. This trajectory is uploaded to the Pixhawk board, and if the mission is not replanned the vehicle will follow the predefined trajectory. Otherwise, if the mission is replanned the quadcopter will follow the new trajectory generated online.

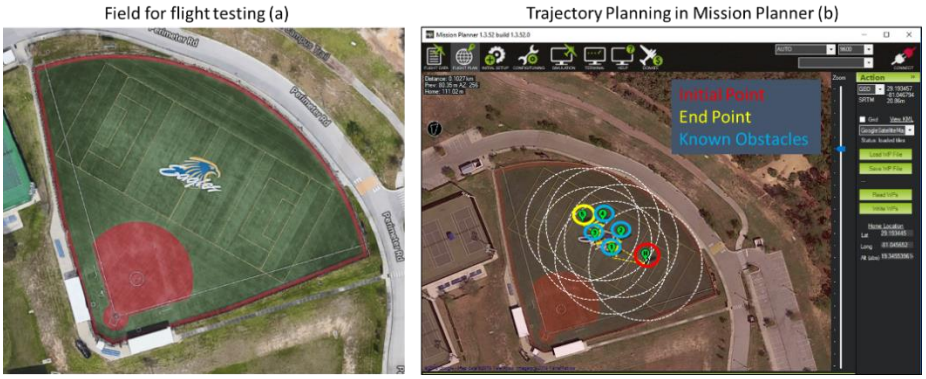

Figure 69 Process for mission execution

The overall procedure for flight testing starts with the take-off of the quadcopter under the control of the pilot using the RC control. After the pilot selects the desired altitude, a switch in the RC control enables the autonomous mission. Once the quadcopter is in self-directed flight, the algorithms dictate all the actions performed. The following sections show results for the response of the intelligent algorithm during flight test under different scenarios.

7.3.1. Unknown Obstacle

Several flight tests have been performed to demonstrate the capabilities of real-time decision making algorithms implemented in a quadcopter test bed. In order to test the decision-making algorithms, the trajectory to accomplish a desired mission is first generated offline as shown in Figure 70. The start point of the trajectory is at $\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 0\ m \\ 0\ m \end{bmatrix}$, represented as a yellow point, and the end point of the trajectory is at $\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 15.85\ m \\ -31.61\ m \end{bmatrix}$ represented as a light blue point. The desired mission is to avoid a known obstacle represented as a magenta circle in Figure 70.

This mission was subject to an unknown threat as shown in Figure 71. The green circle represents the unknown obstacle that is being detected, and from this detection, it is clear that following the original trajectory will cause the vehicle to collide with the obstacle. Thus, decision-making took place and re-planned the original path by selecting and generating a different path to complete the mission safely, shown as the red trajectory in Figure 71. The blue trajectory represents the actual path the vehicle performed. These results show that the capabilities of the vehicle to navigate in an unknown environment are enhanced by the implementation of decision-making algorithms.

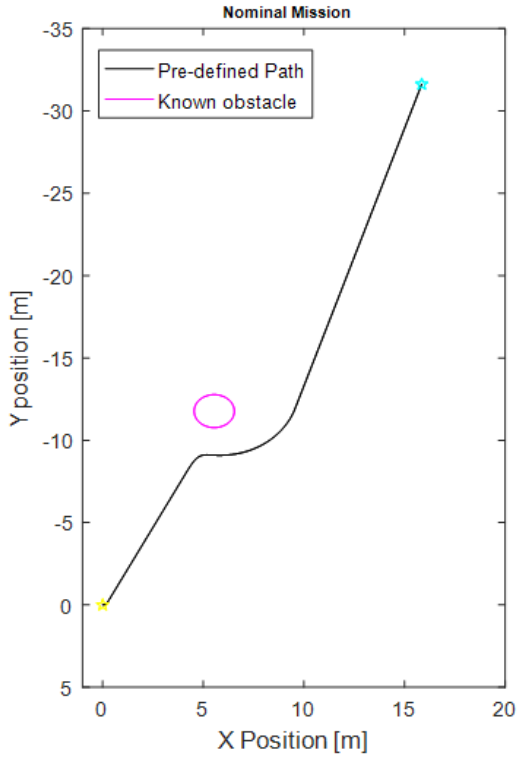

Figure 70 Predefined trajectory for desired mission

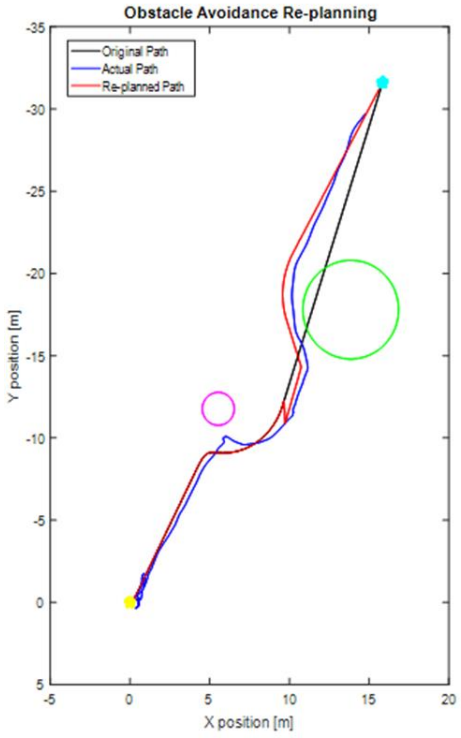

Figure 71 Re-planning capabilities for an unknown obstacle in the mission

To perform more realistic flight tests to test algorithm capabilities, the quadcopter navigated through a simulated urban environment as seen in Figure 72. The positions and radius for the buildings were obtained from the vision system as shown in Figure 73.

Figure 72 Simulated urban environment

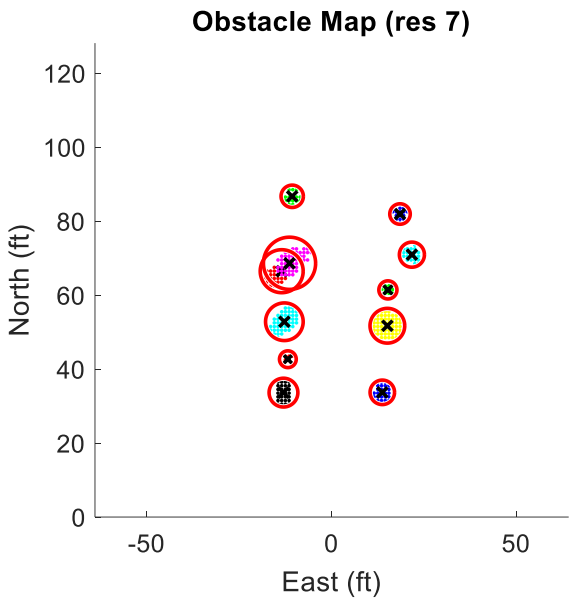

Figure 73 Buildings information from the vision system

With the information from the vision system, a pre-defined trajectory was generated with the potential field algorithm as presented in Figure 74a. A nominal flight was performed with the vehicle through the model city as shown in Figure 74b. Figure 74c shows the re-planning and decision making capabilities of the system under the condition of an unknown obstacle. In the figure, the unknown obstacle is represented as the solid line red circle. It is clear from Figure 74c that if re-planning did not take place, the vehicle would have crashed into the unexpected obstacle encountered. In Figure 74b and Figure 74c the blue trajectory represents the actual path performed by the vehicle, the red trajectory is the commanded path to complete the mission, the green point is the starting point and the light blue point is the end of the trajectory.

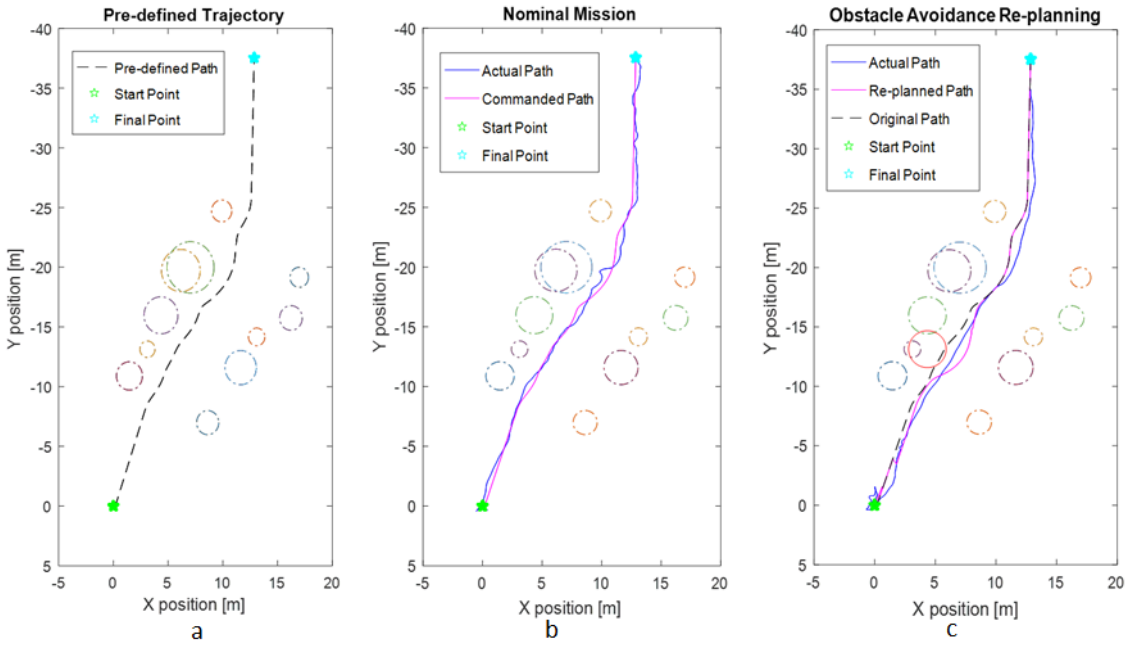

Figure 74 Urban environment navigation

7.3.2. System Failures

The information to re-plan for a failure in the system comes from the health monitoring (HM) capabilities for detection and identification of failures. After the failure is detected and correctly identified, trajectory re-planning takes place. Here the decision-making takes an important step depending on the type of failure present in the system. Through previous simulation analysis of the vehicle dynamics performance, it is known that the vehicle would not behave the same way under failures in different motors.

Several flight tests were performed to prove the capabilities of intelligent algorithms under failure conditions. The results obtained after injecting a failure in motor 1 are shown in Figure 75. Before analyzing the re-planning capabilities and decision-making, it is important to mention the failure in motor 1, as shown in Figure 75c, is being detected by the HM system implemented along with the decision-making algorithms. As shown in Figure 75b, the detectors from the HM system are being activated due to the

detection of a failure. Because the failure is identified in motor 1, the algorithm maintains and generates the same trajectory as the original. This result is expected because as aforementioned, for a failure in motor 1 the algorithm selects and generates trajectories similar to Trajectory 1 as shown in Figure 39. In addition, to ensure the trajectory generated is the most favorable for the vehicle performance, a "virtual" obstacle is calculated depending on the type of failure. The virtual obstacle is illustrated as a green circle, and the actual path performed by the vehicle is represented as the blue trajectory as shown in Figure 75a.

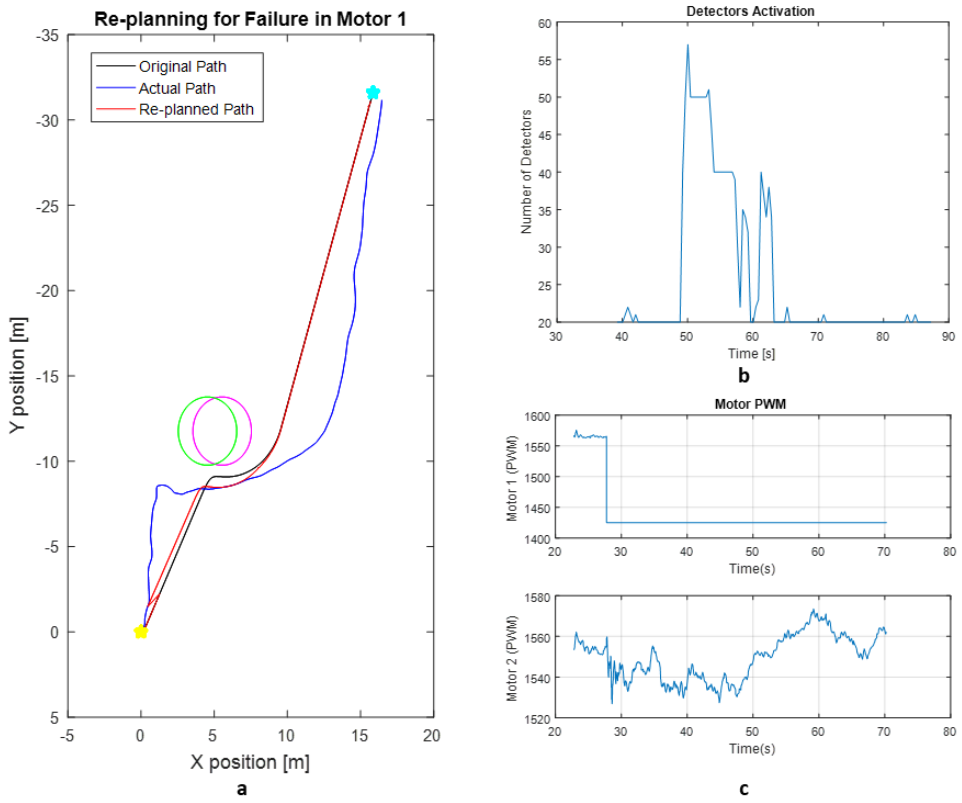

Figure 75 Decision-making and re-planning capabilities for failure in Motor 1

Similarly, results were obtained for failure in motor 2. From the simulation, it is known that if the vehicle presents a failure in motor 2, the decision-making algorithm selects and generates trajectories similar to Trajectory 2 (shown in Figure 39) to assure the

vehicle completes the mission efficiently. The results shown in Figure 76 demonstrate the re-planning potential under failure conditions in motor 2. Similar to previous results the failure, shown in Figure 76c, is being detected by the Health Monitoring system as shown in Figure 76b. The "virtual" obstacle is placed in a different position because the failure is in motor 2, guaranteeing the path generated is secure and the performance is not exacerbated by following the new trajectory. As mentioned before, the green circle represents the "virtual" obstacle and the blue trajectory is the actual path performed by the vehicle during flight test.

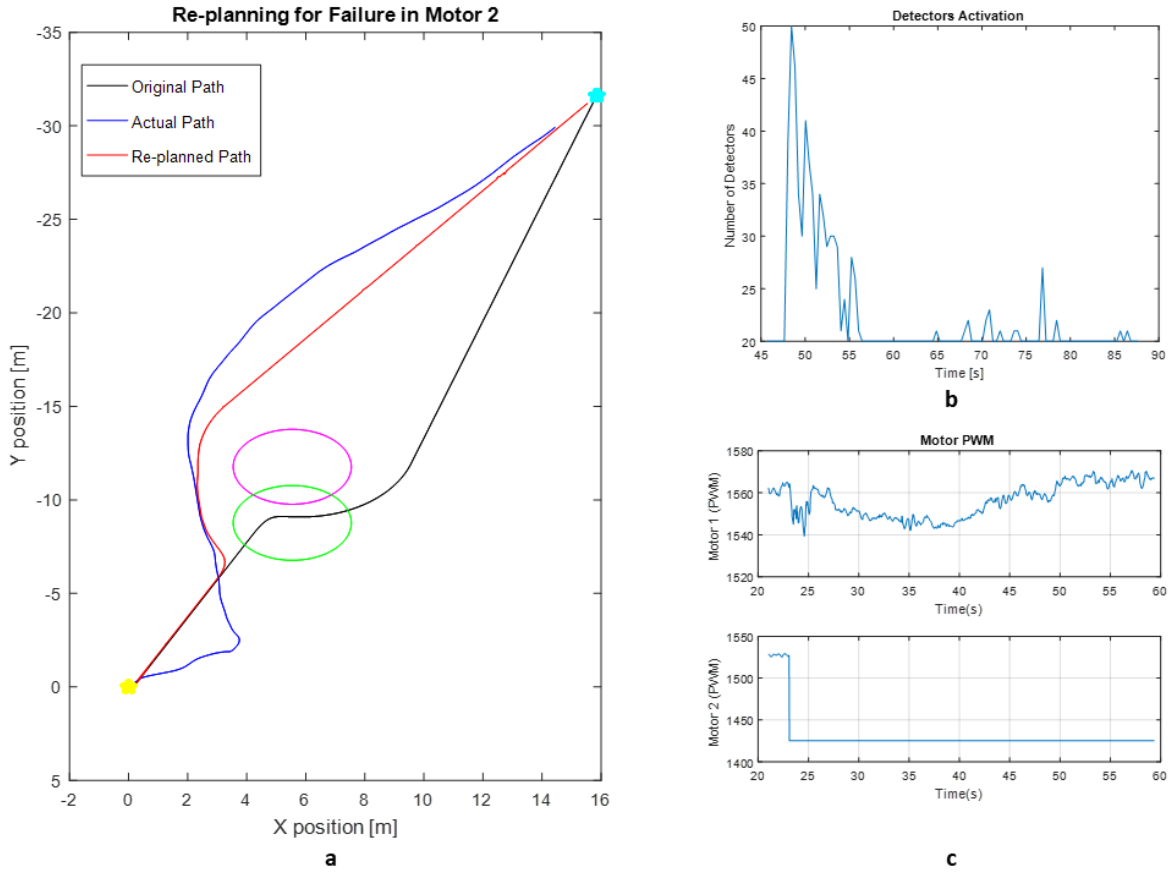

Figure 76 Decision-making and re-planning capabilities for failure in Motor 2

7.3.3. Low Battery Condition

Through the implementation of low battery algorithms, developed in simulation, the quadcopter is able to perform a “return-home” maneuver. Figure 77 shows the results of a “return home” maneuver tested in flight. From the results obtained as shown in Figure 77a, the decision-making algorithms successfully generated the most optimal trajectory to return home while avoiding the known obstacle (red circle) in the mission. The green path represents a predefined path commanded to complete the mission; after low battery was detected by the system, it calculated a new trajectory represented by the yellow path in Figure 77a. As in previous results, the blue path is the actual response from the quadcopter during flight test. This enhances vehicle autonomy and safety for low power conditions. In addition, Figure 77b and c show the low battery trigger and the x-y quadcopter positions respectively.

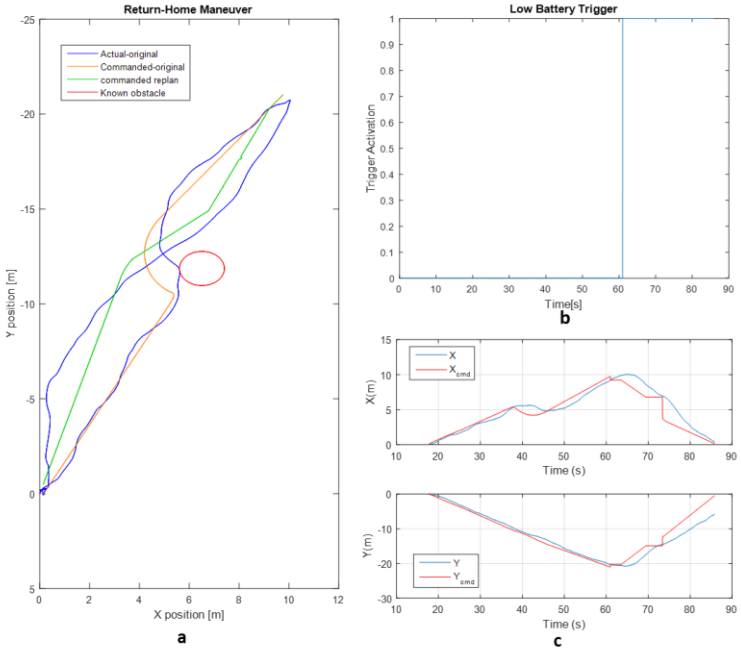

Figure 77 Return-Home maneuver due to low battery

In order to have a better understanding of the vehicle performance during flight, the pilot activates the low battery trigger from the DX8 8CH Transmitter. Figure 78 shows the online estimation of the remaining battery percentage. Decision-making and re-planning algorithms use this information to generate the best solution for the mission.

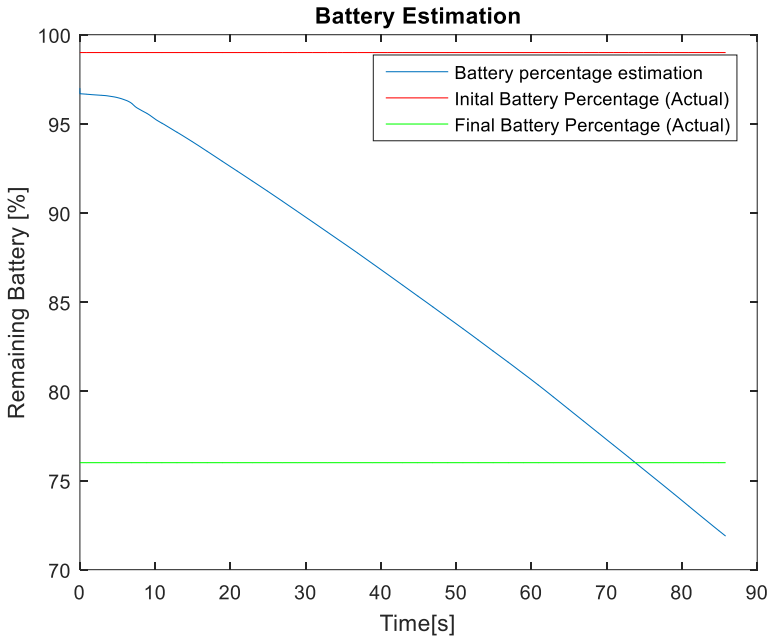

Figure 78 Battery Percentage Estimation

The blue line in Figure 78 represents the online estimation of remaining battery percentage. The red and green limits represent the actual starting and final battery percentage respectively. This estimation improves with more accurate system parameters such as motor draw amps and battery amp hours.

8. Conclusions

An intelligent algorithm has been developed and implemented providing a quadrotor vehicle the capabilities of autonomous decision-making. These capabilities have shown to bolster the performance and efficiency of the mission by planning accordingly with the resources the system possesses such as remaining battery percentage, health, and maneuverability.

Decision-making has been proven to improve mission performance by re-planning the maneuvers after a threat has been encountered. The system performs obstacle avoidance for unknown obstacles, trajectory selection and calculation that is the most favorable when there are failures in the system, and return home safely for low battery conditions.

The intelligent algorithm has been successfully implemented in a quadcopter testbed performing on-line decision-making and on-line re-planning. Having a real-time intelligent algorithm increases the robustness of the vehicle and preserves the integrity of the system and the surrounding environment.

Using the taxonomy presented for intelligent systems the quadcopter system has been proven to be classified in the following categories shown in Table 8:

Table 8 Intelligent System Classification for vehicle developed

PROPERTY	LEVEL	EXPLANATION
METH	SUP	Using offline training for trajectory selection under failures and offline training for health monitoring detectors and identifiers the system is capable of supervised learning
LEARN	PRM	Through the implementation of an adaptive control, the quadcopter learns and changes parameters such as the adaptive gains with respect to the vehicle status, making the system capable of learning new parameters within a given structure of rules
ADAPT	BOTH	The system is capable of failure recovery and obstacle avoidance resulting in a system capable of adapting to both internal and external states

TRIG	REACT	The intelligent algorithm is activated after a threat has been identified, thus the system only adapts as a reaction to state changes after they occur
SCOPE	PRES	The system offers online health monitoring and re-planning. However, there are many constraints to overcome such as online detectors generation, be able to learn new trajectories that benefit the system performance, etc. Thus, the system is capable of adaptation within a certain prescribed set of behaviors
CONS	PERF	The system behaves in the preprogrammed manner but the same behavior in different cases might not result in the same performance. Thus, the system's performance under various conditions is consistent but it may use different behaviors to achieve it
DATA	MULTI	There are many data fusion processes to achieve the required states for autonomous missions such as a Kalman Filter, Estimation, GPS and IMU to name a few. Therefore, the system has several modalities of information sources.
FUSE	MULTI	The system performs some data fusion levels such as estimation, resource management but does not perform predictions of impact. Thus, the system performs several but not all levels of data fusion.
MANI	ENTITY	For this thesis, only one vehicle was used for simulation and implementation of intelligent algorithms. Thus, the manifestation of intelligent system properties is at the entity level. COLLC and HYBRID systems are considered swarms or groups of intelligent robots working together. (Sankararaman,2017)

9. Future Work & Recommendations

For this thesis, only the Potential Field algorithm has been implemented within the intelligent algorithm. However, the Clothoid algorithm can also be implemented along with the Potential Field to perform missions not only for obstacle avoidance but for waypoint navigation as well.

One of the main limitations of the algorithms developed in this thesis is that they work for a 2D environment. In all the simulations and implementation tests, the altitude is maintained constant. These intelligent algorithms can be expanded to 3D dimension providing the quadcopter with a wide range for maneuverability and mission goals. Furthermore, 3D algorithms can also improve the efficiency of the trajectory generation. For some circumstances, changing altitude will be more favorable than changing direction in the x-y plane.

In addition, the system could be improved by enabling online training. For some of the calculations in the decision making processes such as health monitoring of the system, or the generation of virtual obstacles for failures in the system. These processes should be enhanced for real-time operation.

The battery estimation model presented in this thesis can be enhanced by implementing either a better estimation model based on voltage measurement instead of current, or a sensor that measures the remaining voltage in the battery.

The NLDI controller can be polished, expanding its capabilities for wind rejection. In addition, the wind can be considered as a threat for the intelligent algorithm.

For this thesis, the position of unknown obstacles has been designed to be random.

For future work complex navigation environments should be consider. Such as organized forest for search and rescue missions.

The last but not least, the experiments performed in this thesis consider one threat per mission. To increase the proposed algorithm robustness, reliability, and autonomy, missions with threat combinations must be studied.

REFERENCES

Allen, R., Pavone, M. (2016). “A Real-Time Framework for Kinodynamic Planning with Application to Quadrotor Obstacle Avoidance”. *AIAA Guidance Navigation and Control Conference, SciTech Forum.* San Diego, California.

Atkins, E. (2012). “Intelligent Systems for Unmanned Aircraft Safety Certification”. *AIAA Aerosapce Sciences Meeting including the New Horizons Forum and Aerospce Exposition.* Nashville, Tennessee.

Bhaduri, R. (2009). “A Mobile Robot Path Planning Using Genetic Artificial Immune Network Algorithm”. *IEEE World Congress on Nature & Biologically Inspired Computing.* India.

Boskovic, J., Prasanth, R., Mehra, R. (2004). “A Multi-Layer Autonomous Intelligent Control Architecture for Unmanned Aerial Vehicles”. *Journal of Aerospace Computing, Information, and Communication*, Vol.1.

Cesare, K., Skeele, R., Yoo, S., Zhang, Y., Hollinger, G. (2015). “Multi-UAV Exploration with Limited Communication and Battery”. *IEEE International Conference on Robotics and Automation (ICRA),* Seattle, Washington.

Cetin, O., Yilmaz, G. (2014). “Sigmoid limiting functions and potential field based autonomous air refueling path planning for UAVs”. *Journal of Intelligent & Robotic Systems*, *73*(1-4), 797-810.

Cetin, O., Yilmaz, G. (2016). “Real-time autonomous UAV formation flight with collision and obstacle avoidance in unknown environment”. *Journal of Intelligent & Robotic Systems*. 84(1-4), 415-433.

Chamseddine, A., Theilliol, D., Zhang, Y.M., Join, C., Rabbath, C.A. (2015). “Active fault-tolerant control system design with trajectory re-planning against actuator faults and saturation: Application to quadrotor unmanned aerial vehicle”. *International Journal of Adaptive Control and Signal Processing.* Vol.29. Issue 1. Pages 1-23.

Chang, K., Xia, Y., Huang, K., Ma, D. (2016). “Obstacle avoidance and active disturbance rejection control for a quadrotor”. *ELSEVIER Neurocomputing.*

Colgren, R. (2007). Introduction to Stateflow, Basic MATLAB, Simulink, and Stateflow. AIAA Education Series, pp 367-424.

Cruz, G. C., Santos, Encarnação, P. M., Martins. (2012). “Obstacle avoidance for unmanned aerial vehicles”. *Journal of Intelligent & Robotic Systems*. 65(1-4), 203-217.

Das, A., Subbarao, K., Lewis, F. (2008). “Dynamic inversion with zero-dynamics

stabilisation for quadrotor control". *IET Control Theory and Applications.*

DeGarmo, M. (2004). "Issues Concerning Integration of Unmanned Aerial Vehicles in Civil Aerospace". *MITRE Center for Advanced Aviation System Development*, McLean, Virginia.

Garcia, D. (2017). "Design, Development and Implementation of Intelligent Algorithms to Increase Autonomy of Quadrotor Unmanned Mission (Master Thesis)". Department of Aerospace Engineering. Embry-Riddle Aeronautical University.

Garcia, D., Moncayo, H., Perez, A., Rivera, K., Dupuis, M., and Mueller, R. "Spacecraft Health Monitoring Using a Biomimetic Fault Diagnosis Scheme", *AIAA Information Systems-AIAA Infotech @ Aerospace*. AIAA SciTech Forum, (AIAA 2017-1294).

Ghaffar, A., Richardson, T. (2016). "Position Tracking of an Underactuated Quadrotor using Model Reference Adaptive Control". *AIAA Guidance Navigation and Control Conference, SciTech Forum.* San Diego, California.

Hernandez, D., Vidal, E., Greer, J., Fiasco, R., Jaussaud, P., Carreras, M., Garcia, R., (2017). "AUV online mission replanning for gap filling and target inspection". *OCEANS.*

Hongwei Mo. (2009). "Artificial Immune Systems and Natural Computing ed". Medical Information S*cience Reference.* New York, Chapter XV.

Hwang, Y., Ahuja, N. (1992). "A Potential Field Approach to Path Planning". *IEEE Transactions on Robotics and Automation*. Vol.8. No. 1, pp. 23-32.

Ireland, M., Vargas, A., Anderson, D. (2015). "A Comparison of Closed-Loop Performance of Multirotor Configurations Using Non-Linear Dynamic Inversion Control". *Aerospace*, 2(2), 325-352. doi:http://dx.doi.org.ezproxy.libproxy.db.erau.edu/10.3390/aerospace2020325.

Jeyarman, Suresh, et al. (2005). "Formal Techniques for the Modeling and Validation of a Co-operating UAV Team that uses Dubins Set for Path Planning". *Proceedings of the 2005 American Control Conference*. pp. 4690-4695.

Jones, P., Ludington, B., Reiman, J., Vachtsevanos, G. (2006). "Intelligent Control of Unmanned Aerial Vehicles for Improved Autonomy". *European Journal of Control*. Vol.13. Issues 2-3. Pages 320-333.

Jung, D., Parker, L. (2002). "Path Planning is no substitute for intelligent behavior". *Proc. SPIE 4715, Unmanned Ground Vehicle Technology IV*.

Kaneshige, J., Krishnakumar, K. (2005). "Tactical Immunized Maneuvering System for Exploration Air Vehicles". *Infotech Aerospace Conferences. NASA Ames*

Research Center. California.

Kaneshige, J., Krishnakumar, K. (2007). "Artificial Immune System Approach for Air Combat Maneuvering". *Proc. SPIE 6560, Intelligent Computing: Theory and Applications V*.

Kikutis, R., Stankunas, J., Rudinskas, D., Masiulionis, T. (2017). "Adaptation of Dubins Paths for UAV Ground Obstacle Avoidance When Using a Low Cost On-Board GNSS Sensor". *Sensors, 17(10), 2223*.

Kivelevitch, E. (2015). "A Taxonomy of Intelligent Systems". *AIAA SciTech Forum*, Kissimmee, Florida.

Krishnakumar, K., Kopardekar, P., Ippolito, C., Melton, J., Stepanyan, V., Sankararaman, S., Nikaido, B. (2012). "M-MRAC for Nonlinear Systems with Bounded Disturbances". NASA Ames Research Center. *50th IEEE Conference on Decision and Control*.

Krishnakumar, K., Kopardekar, P., Ippolito, C., Melton, J., Stepanyan, V., Sankararaman, S., Nikaido, B. (2017). "Safe Autonomous Flight Environment (SAFE50) for the Notional Last 50 ft of Operation of 55 lb Class of UAS". *NASA Ames Research Center*. ARC-E-DAA-TN38504.

Krishnakumar, K., Lohn, J., Kaneshige, J. (2004). "Intelligent Systems: Shaping the Future of Aeronautics and Space Exploration". *NASA Ames Research Center.* RTOP 704-30-62.

Krishnakumar,K. "Intelligent Systems for Aerospace Engineering—An Overview". NeuroEngineering Laboratory. NASA Ames Research Center.

Krishnakumar,K., Kanashige,J. (2002). "Challenging Aerospace Problems for Intelligent Systems". *Intelligent Systems for Aeronautics*, RTO-EN-022.

Lin, L., Peng, J., Fan, Y., Liu, Y. (2010). "Path Optimization Algorithm for Agents based on Artificial Immune and Emotional Learning". *IEEE International Conference on Control and Automation,* China.

Min, Y., Min, Z. (2015). "Unmanned aerial vehicle dynamic path planning in an uncertain environment". *Robotica, 33(3), 611-621*.

Moncayo, H., Perhinschi, M., Davis, J. (2010). "Aircraft Failure Detection and Identification Using an Immunological Hierachical Multoself Strategy". *Journal of Guidance, Control, and Dynamics.* Vol 33. No. 4.

Moncayo, H., Perhinschi, M., Wilburn, B., Wilburn, J., Karas, O. (2012). "Extended Nonlinear Dynamic Inversion Control Laws for Unmanned Air Vehicles". *AIAA Guidance, Navigation, and Control Conference.* Minneapolis, Minnesota.

Montes, N., Mora, M., Tornero, J. (2007). "Trajectory Generation based on Rational Bezier Curves as Clothoids". *IEEE Intelligent Vehicles Symposium*. pp. 505-510.

Moon, S., Oh, E., Shim, D. (2013). "An Integral Framework of Task Assignment and Path Planning for Multiple Inmanned Aerial Vehicles in Dynamic Environments". Journal of *Intelligent & Robotic Systems,* Tomo.70. No.1-4.

Nguyen, N., Krishnakumar, K., Boskovic, J. (2008). "An Optimal Control Modification to Model-Reference Adaptive Control for Fast Adaptation". NASA Ames Research Center. *AIAA Guidance, Nvigation and Control Conference*.

Nguyen, N., Krishnakumar, K., Kaneshige, J., Neespeca, P. (2006). "Dynamics and Adaptive Control for Stability Recovery of Damaged Aircraft". NASA Ames Research Center. *AIAA Guidance, Nvigation and Control Conference*.

Nguyen, N., Krishnakumar,K. (2009). "Hybrid Intelligent Flight Control with Adaptive Learning Parameter Estimation". *Journal of Aerospace Computing, Information, and Communication.* Vol.6. No.3. pp. 171-186.

Ojha, U., Chow, M. (2010). "An Analysis of Artificial Immune System and Genetic Algorithm in Urban Path Planning". *IEEE Annual Conference on IEEE Industrial Electronics Society*. Arizona, USA.

Paul, T., Krogstad, T., Gravdahl, J.T. (2008). "Modelling of UAV formation flight using 3D potential field". *ELSEVIER Simulation Modelling Practice and Theory*.

Perez, A. (2016). "Development of fault tolerant adaptive control laws for aerospace systems (Doctoral Dissertation)". Department of Aerospace Engineering. Embry-Riddle Aeronautical University.

Perez, A., Moncayo, H., Perhinschi, M., Al Azzawi, D., Togayev, A. (2015). "A Bio-Inspired Adaptive Control Compensation System for an Aircraft Outside Bounds of Nominal Design". *Journal of Dynamic Systems, Measurements and Control, ASME*. Vol.137.

Perez, A.E., Moncayo, H., Perhinschi, M., Al Azzawi, D., & Togayev, A. (2015). "A bio-inspired adaptive control compensation system for an aircraft outside bounds of nominal design". *Journal of Dynamics Systems, Measurements, and Control*, Vol.137. Issue 9.

Perhinschi, M., Moncayo, H., Dvis, Jennifer. (2010). "Integrated Framework for Artificial Immunity-Based Aircraft Failure Detection, Identification, and Evaluation". *Journal of Aircraft.* Vol 47. No. 6.

Pixhawk Support Package User Guide from Mathworks, 2017.

Pu, H., Zhen, Z., Jiang, J., Wang, D. (2013). "UAV Flight Control System Based on an

Intelligent BEL Algorithm". *International Journal of Advanced Robotic Systems*. Vol.10. Issue 2.

Ramirez, C., Bello, G., Moreno, M., Camacho, D. (2016). "MOGAMAR: A Multi-Objective Genetic Algorithm for real-time Mission Replanning". *IEEE Symposium Series on Computational Intelligence (SCCI).*

Sankararaman, S., Krishnakumar, K. (2017). "Towards A Computational Framework for Autonomous Decision-Making in Unmanned Aerial Vehicles". *AIAA Information Systems-AiAA SciTech Forum*. NASA Ames Research Center.

Scheuer, A., Fraichard, T. (1997). "Continuous-Curvature Path Planning for Car-Like Vehicles". *IEEE International Conference on Intelligent Robots and Systems*. pp. 997-1003.

Shanmugavel, Madhavan, et al. (2010). "Co-operative Path Planning of Multiple UAVs Using Dubins Path with Clothoid Arcs". *ELSEVIER ScienceDirect Control Engineering Practice*. Vol.18.

Tsourdos, A., White, B., Shanmugavel, M. (2011). "Cooperative Path Planning of Unmanned Aerial Vehicles". *AIAA Wiley*. Vol.235, 2, pp. 29-36.

Wilburn, J, Perhinschi, M., Wilburn, B. (2013). "Implementation of a 3-Dimensional Dubins-Based UAV Path Generation Algorithm". *AIAA Guidance, Navigation, and Control Conference*.

Wilburn, J, Perhinschi, M., Wilburn, B. (2013). "Implementation of Composite Clothoid Paths for Continuous Curvature Trajectory Generation for UAVs". *AIAA Guidance, Navigation, and Control*. Boston, MA.

Wilburn, J. (2013. "Development of an Integrated Intelligent Multi-Objective Framework for UAV Trajectory Generation (Doctoral Dissertation)".Department of Mechanical and Aerospace Engineering. West Virginia University.

Yao, P., Wang, H., Su, Z. (2015). "Real-time path planning of unmanned aerial vehicle for target tracking and obstacle avoidance in complex dynamic environment". *Aerospace Science and Technology*, Vol.27. Pages 269-279.

Yuanchen, Z., Lu, J., Rui, Z., Jie, Z. (2017). "UAV formation control with obstacle avoidance using improved artificial potential fields". *IEEE Chinese Control Conference (CCC)*. China.